

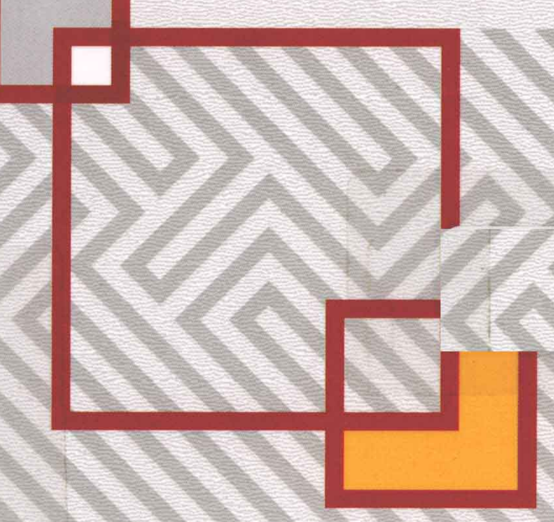
21世纪高等院校信息与通信工程规划教材

21st Century University Planned Textbooks of Information and Communication Engineering

现代电子设计技术与综合应用

成谢锋 孙科学 张学军 编著

The Technology of Modern Electronic Design







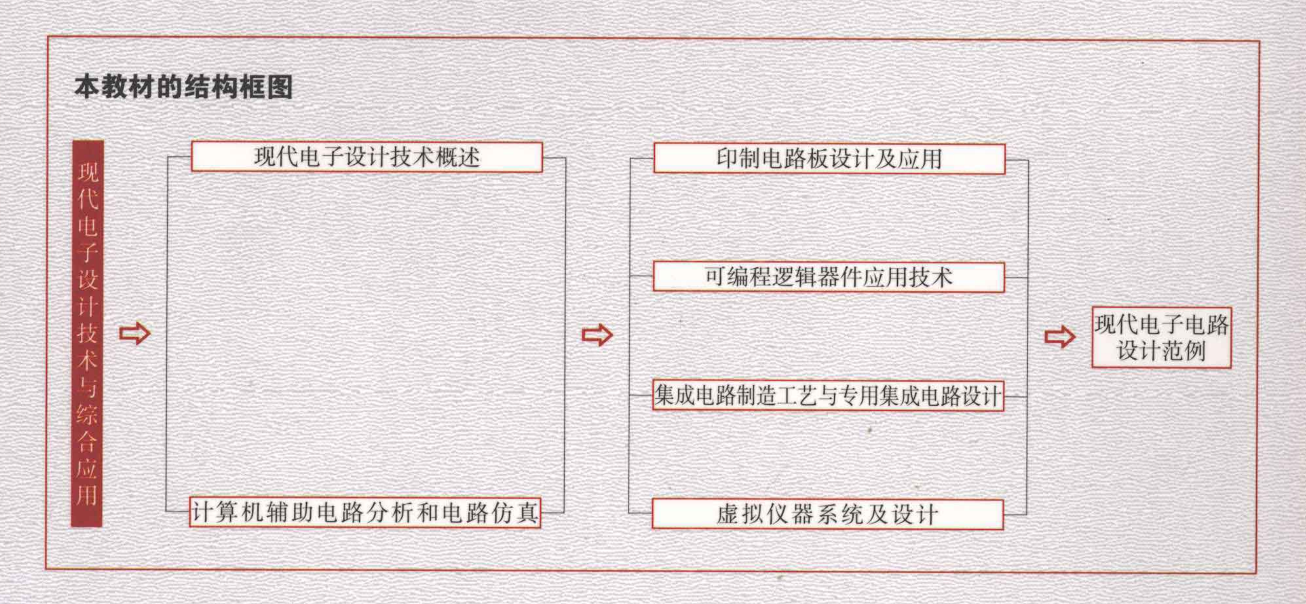
精品系列

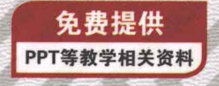
The Technology of Modern Electronic Design

现代电子设计技术 与综合应用

本书从实用的角度出发，介绍了现代电子设计技术的基本理论和方法。全书共7章，内容包括现代电子设计技术概述、计算机辅助电路分析和电路仿真技术、印制电路板设计及应用、可编辑逻辑器件应用技术、集成电路制造工艺与专用集成电路设计、虚拟仪器系统及设计和现代电子电路设计范例。

本书在内容上深入浅出，注重实用性，兼顾理论教学和自学的需求，配备了大量的应用实例，使读者能在较短时间内掌握现代电子设计技术的基本理论和方法。本书既可作为高等学校电子设计技术课程的教材，也可作为电子系统开发人员的技术参考书。







教材服务热线：010-67170985

人民邮电出版社教学服务与资源网：www.ptpedu.com.cn

封面设计：董志桢

人民邮电出版社网址：www.ptpress.com.cn





ISBN 978-7-115-24662-2

定价：28.00 元

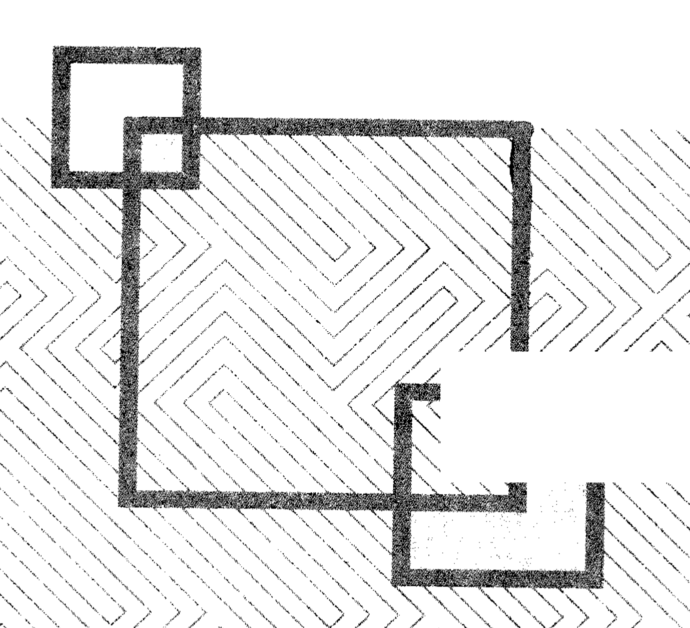
21世纪高等院校信息与通信工程规划教材

21st Century University Planned Textbooks of Information and Communication Engineering

现代电子设计技术与综合应用

成谢锋 孙科学 张学军 编著

工业和信息化普通高等教育“十二五”规划教材立项项目



人民邮电出版社
北京



图书在版编目（CIP）数据

现代电子设计技术与综合应用 / 成谢锋，孙科学，
张学军编著. -- 北京 : 人民邮电出版社，2011.3
21世纪高等院校信息与通信工程规划教材
ISBN 978-7-115-24662-2

Ⅰ. ①现… Ⅱ. ①成… ②孙… ③张… Ⅲ. ①电子电
路—电路设计—高等学校—教材 Ⅳ. ①TN702

中国版本图书馆CIP数据核字(2011)第002254号

内 容 提 要

本书从实用的角度出发，介绍了现代电子设计技术的基本理论和方法。全书共7章，内容包括现代电子设计技术概述、计算机辅助电路分析和电路仿真技术、印制电路板设计及应用、可编辑逻辑器件应用技术、集成电路制造工艺与专用集成电路设计、虚拟仪器系统及设计和现代电子电路设计范例。

本书在内容上深入浅出，注重实用性，兼顾理论教学和自学的需求，配备了大量的应用实例，使读者能在较短时间内掌握现代电子设计技术的基本理论和方法。本书既可作为高等学校电子设计技术课程的教材，也可作为电子系统开发人员的技术参考书。

21世纪高等院校信息与通信工程规划教材

现代电子设计技术与综合应用

◆ 编　　著　成谢锋　孙科学　张学军
　责任编辑　蒋　亮

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街14号
　邮编　100061　　电子函件　315@ptpress.com cn
　网址　http //www ptpress com cn
　北京昌平百善印刷厂印刷

◆ 开本：787×1092　　1/16
　印张：14 75　　　　2011年3月第1版
　字数：362千字　　　2011年3月北京第1次印刷

ISBN 978-7-115-24662-2

定价：28.00元

读者服务热线：(010)67170985　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前言

现代电子设计技术由于计算机技术的快速发展而不断发生着巨大变化。经典的电子设计方法，是用电路图表示设计思想，用实验电路板搭载实验电路，进行模拟、仿真，用电子测试仪器进行功能、性能测试。从20世纪60年代中期开始，人们不断开发出各种计算机辅助设计工具来帮助设计人员进行集成电路和电子系统的设计，集成电路技术的不断发展对电子设计技术提出新的要求，使得电子设计技术很快由计算机辅助设计（CAD）阶段进入了电子设计自动化（EDA）阶段。这是一个质的飞跃，因为，在EDA中，用硬件描述语言表达设计思想，用计算机进行模拟、仿真，并把测试器件设计到芯片系统内部，实现了内建自测试功能。利用先进的计算机工作平台开发出的一整套电子系统设计软件工具，实现电子产品从电路设计、性能分析到设计出IC版图或PCB版图的整个过程的自动处理以及其他利用计算机平台实现电子产品设计的技术，我们统称之为现代电子设计技术，它是科研、教学领域的一门新兴的工程技术，也是电子信息行业的一项先进的生产技术。

本书是配合江苏省省级精品课程《电工电子实验》的教学改革而编写的。通过本教材展现现代电子设计技术的发展过程，论述南京邮电大学省级电工电子实验中心EDA教学改革的一些成果，促进学生掌握电子电路计算机辅助分析与自动化设计的基本知识和基本方法，进一步培养学生的综合应用能力和实践能力，为今后从事本专业有关工程技术工作打下基础。

本书从实用的角度出发，介绍现代电子设计技术的基本理论和方法。全书共分7章。第1章讲述现代电子设计技术概述；第2章讲述计算机辅助电路分析和电路仿真技术；第3章讲述印制电路板设计及应用；第4章讲述可编程逻辑器件应用技术；第5章讲述集成电路制造工艺与专用集成电路设计；第6章讲述虚拟仪器系统及设计；第7章讲述现代电子电路设计范例。本书在内容上深入浅出，注重实用性，兼顾理论教学和自学的需求，配备了大量的应用实例，使读者能在较短的时间内掌握现代电子设计技术的基本理论和方法。

操作显能力，设计见智慧。在电子产品设计过程中不仅需要较多的知识，还需要有丰富的设计实践经验和先进的设计理念，否则在设计过程中很难找到切实可行又有竞争力的方案。所以本书中既有一般的应用实例，又加入了一些我们团队自己设计的方案。这些实例都是在相应的软件平台上调试运行通过的，读者可以参照书中所讲述的步骤操作，顺利完成学习任务，全面提高综合应用能力和设计技能。作为教材，本书每章后还附有习题。

参加本书编写有关工作的老师和同学还有南京邮电大学赵青、朱冬梅、何海琴、曹刚、

吴健、王路飞和蚌埠学院孙长伟（第6章）等。本书的顺利出版，要感谢南京邮电大学的领导、电工电子实验中心的教师和人民邮电出版社给予的大力支持和帮助。

由于时间仓促，书中难免存在不妥之处，恳请读者多提宝贵意见。

作者电子信箱：chengxf@njupt.edu.cn

作 者

2010年12月于南京邮电大学

目　　录

第 1 章 现代电子设计技术概述

本章要点

- 电子设计自动化（EDA）的基本概念
- EDA 技术的发展历程
- 电子电路设计的一般方法
- EDA 技术的基本特征
- 本书所选用的 EDA 软件的简介

1.1 电子设计技术的发展历程

EDA 技术是一种实现电子系统或电子产品设计自动化的技术，与电子技术、微电子技术的发展密切相关，它是以计算机为基本工作平台，利用计算机图形学、图论与拓扑逻辑、计算数学、优化理论等多学科最新成果研制出的计算机辅助设计通用软件工具。EDA 技术是一种帮助电子设计工程师从事电子组件产品和系统设计的综合技术。EDA 是在 20 世纪 90 年代从计算机辅助设计（CAD）、计算机辅助制造（CAM）、计算机辅助测试（CAT）和计算机辅助工程（CAE）的概念发展而来的。一般把 EDA 技术的发展分为 CAD、CAE、EDA 3 个阶段。

20 世纪 70 年代的 CAD 是 EDA 技术发展的早期阶段。在这个阶段，人们开始利用计算机取代手工劳动，但当时的计算机硬件功能有限，软件功能较弱，人们主要借助计算机对所设计的电路进行一些模拟和预测，辅助进行集成电路版图编辑和印刷电路板（PCB）布局、布线等简单的版图绘制等工作。这个时期受到计算机工作平台的限制，能支持的设计工作有限且性能比较差，效率较低。

20 世纪 80 年代的 CAE 是在 CAD 工具逐步完善的基础上发展起来的，即所谓的 EDA 技术中级阶段。其主要特征是具备了自动布局布线和电路的计算机仿真、分析和验证功能。其作用已不仅仅是辅助设计，而且可以代替人进行某种思维。CAE 这种以原理图为基础的 EDA 系统，虽然直观，且易于理解，但对复杂的电子设计很难达到要求，也不宜于设计的优化。

20 世纪 90 年代以来，微电子工艺有了快速的发展，工艺水平达到了深亚微米级，甚至达到超深亚微米级。在一个芯片上已经可以集成上百万乃至上亿只晶体管，芯片速度达到了 Gbit/s 量级，百万门以上的可编程逻辑器件面世。在这种形势下，EDA 技术的发展得到了极

大的推动。EDA 技术理论更加成熟，EDA 工具软件具备了更多的功能、更高的速度及更高的自动化程度。这都极大地提高了系统设计的效率，缩短了产品的研制周期，推动了全新的电子设计自动化技术的发展。

现代电子设计技术由于计算机技术的快速发展而不断发生着巨大变化。今天，EDA 技术已经成为现代电子设计的重要工具，无论是芯片设计还是系统设计，如果没有 EDA 工具的支持，都将是难以完成的。

这些利用先进的计算机工作平台所开发出的一整套电子系统设计软件工具，实现了电子产品从电路设计、性能分析到设计出 IC 版图或 PCB 版图的整个过程的自动处理以及其他利用计算机平台所实现电子产品设计自动化的技术，我们统称之为现代电子设计技术，它是科研、教学领域的一门新兴的工程技术，是电子信息行业的一项先进设计生产技术。

1.2 电子电路设计的一般方法

1.2.1 模拟电路的设计方法

模拟电子系统种类繁多、千差万别，设计一个模拟电子系统的方法和步骤也不尽相同。但对于要设计的实际电子系统，一般首先根据电子系统的设计任务，进行总体方案选择；然后对组成系统的单元电路进行设计、参数计算、元器件的确定和实验调试；最后绘出用于指导工程的电路图。

1. 总体方案确定

在全面分析电子系统任务书所下达的系统功能、技术指标后，根据已掌握的知识和资料，将总体系统按功能合理地分解成若干个子系统（单元电路），并画出由各个单元电路框图相互连接而形成的系统原理框图。电子系统总体方案的选择，将直接决定电子系统设计的质量，因此，在进行总体方案设计时，要多思考、多分析、多比较。要从性能稳定性、工作可靠性、电路结构、成本、功耗、调试、维修等方面，选出最佳方案。

2. 单元电路设计

在进行单元电路设计时，必须明确对各单元电路的具体要求，详细拟定出单元电路的性能指标，认真考虑各单元之间的相互联系，注意前后级单元之间信号的传递方式和匹配，尽量少用或不用电平转换之类的接口电路，并应使各单元电路的供电电源尽可能的统一，以使整个电子系统简单可靠。另外，应尽量选择现有的、成熟的电路来实现单元电路的功能。如果找不到完全满足要求的现成电路，则可以在与设计要求比较接近的电路基础上作适当改进，或自己进行创造性设计。为使电子系统的体积小、可靠性高，单元电路尽可能用集成电路组成。

3. 参数计算

在进行电子电路设计时，应根据电路的性能指标要求决定电路元器件的参数。例如，根据电压放大倍数的大小，可决定反馈电阻的取值；根据振荡器要求的振荡频率，利用公式可算出决定振荡频率的电阻和电容值等。但一般满足电路性能指标要求的理论参数值不是唯一

的，设计者应根据元器件的性能、价格、体积、通用性和货源等方面灵活选择。计算电路参数时应注意以下几点：

（1）在计算元器件工作电流、电压和功率等参数时，应考虑工作条件最不利的情况，并留有适当的余量。对于元器件的极限参数必须保留足够的余量，一般取1.5～2倍的额定值。

（2）对于电阻、电容参数的取值，应选计数值附近的标称值。

（3）在保证电路达到性能指标要求的前提下，尽量减少元器件的品种、价格及体积等。

4．元器件选择

在确定电子元器件时，应全面考虑电路处理信号的频率范围、环境温度、空间大小、成本高低等诸多因素。

（1）一般优先选择集成电路。由于集成电路体积小、功能强，能使电子电路可靠性增强，调试方便，并可大大简化电子电路的设计。随着模拟集成技术的不断发展，适用于各种场合下的集成运算放大器不断涌现，只要外加少量的元器件，利用运算放大器就可构成性能良好的放大器。同样，目前在进行直流稳压电源设计时，已很少采用分立元器件进行设计了，取而代之的是性能更稳定、工作更可靠、成本更低廉的集成稳压器。

（2）正确选择电阻器和电容器。这是两种最常见的元器件，种类很多，性能相差很大，应用场合也不同。因此，对于设计者来说，应熟悉各种电阻器和电容器的主要性能指标和特点，以便根据电路要求，对元件做出正确的选择。

（3）选择分立半导体元件。首先要熟悉这些元件的性能，掌握它们的应用范围；再根据电路的功能要求和元器件在电路中的工作条件，如通过的最大电流、最大反向工作电压、最大工作频率、最大消耗功率等，确定元器件的型号。

5．计算机模拟仿真

随着计算机技术的飞速发展，电子系统的设计方法发生了很大的变化。目前，EDA技术已成为现代电子系统设计的必要手段。在计算机平台上，利用EDA软件可对各种电子电路进行调试、测量、修改，大大提高电子设计的效率和精确度，同时节约了设计费用。

6．实验

电子设计要考虑的因素和问题相当多，由于电路在计算机上进行模拟时所采用的元器件参数和模型与实际器件有差别，所以对经计算机仿真过的电路，还要进行实际实验。通过实验可以发现问题、解决问题。若性能指标达不到要求，应深入分析问题出在哪些单元电路或电子器件上，再对它们重新设计和选择，直到性能指标完全满足要求为止。

7．总体电路图绘制

总体电路图是在总框图、单元电路设计、参数计算和元器件选择的基础上绘制的，它是组装、调试、印刷电路板设计和维修的依据。目前一般是利用绘图软件绘制电路图。绘制电路图时要注意以下几点。

（1）总体电路图要尽可能画在同一张图上，同时注意信号的流向，一般从输入端画起，由左至右或由上至下按信号的流向依次画出各单元电路图。对于电路图比较复杂的，应将主

电路图画在一张或数张图纸上，并在各子图所有端口的两端标注上标号，依次说明各图纸之间的连线关系。

（2）注意总体电路图的紧凑，要求布局合理、排列均匀。图中元器件的符号应标准化，元件符号旁边应标出型号和参数。集成电路通常用方框表示，在方框内标出它的型号，在方框的两侧标出每根连线的功能和管脚号。

（3）连线一般画成水平线或垂直线，并尽可能减少交叉和拐弯。对于相互交叉的线，应在交叉处用圆点标出。对于连接电源负极的连线，一般用接地符号表示；对于连接电源正极的连线，仅需标出电压值。

1.2.2 数字电路的设计方法

数字系统由组合逻辑和时序逻辑功能器件组成，而这些功能器件又可以由各种各样的 SSI（小规模集成电路）、MSI（中规模集成电路）、LSI（大规模集成电路）和 VLSI（超大规模集成电路）器件组成。数字系统的设计方法分为传统的“自下而上”和现代的“自上而下”两种设计方法。

1. 数字系统“自下而上”的设计方法（积木方式）

数字系统“自下而上”的设计方法的基本思想是先把系统的总体方案分成若干个相互独立的功能电路，然后用组合逻辑电路和时序逻辑电路的设计方法分别设计并构成这些功能电路，或者直接选择合适的 SSI、MSI、LSI 器件实现上述功能，最后把这些已经确定的功能器件按要求拼接组合起来，便构成完整的数字系统。这种方法适用于规模不大、功能不复杂的数字系统。设计方法的基本步骤如下。

（1）分析任务要求，确定总体方案。根据设计任务书，明确系统的逻辑功能，数据的输入输出方式，系统需要完成的任务等，选定实现系统功能所要遵循的原理和方法。

（2）划分逻辑单元，画出其原理框图。根据数字系统的总体功能，把一个较复杂的逻辑电路分解为若干个较简单的单元电路，明确各个单元的功能及作用，画出其原理框图。逻辑单元大小要适当，以功能单一（便于调测）、易于实现且便于进行方案比较为原则。

（3）选择集成器件类型、确定单元电路的组成。按照每个单元电路的逻辑功能，选择合适的集成器件代替。器件的选择应尽量选用 MSI 和 LSI，由于器件类型和性能的不同，需要器件的数量和连接方式不一样，所以需将不同方案进行比较。一般情况下，选择性能可靠、使用器件少、成本低、器件易于获得的方案。

（4）考虑单元电路的连接。各单元电路选定之后，还要认真解决它们之间的连接问题，如前级对后级是否能驱动，要保证各单元电路之间在时序上一致，并能稳定工作，避免出现竞争冒险或相互之间的干扰。

（5）画出系统框图和逻辑电路图。框图要求能简明扼要地反映系统的工作过程和工作原理，要求能清晰地表示出控制信号和数字信号的流动方向。

逻辑电路图除采用手工绘图外，也可以采用 Protel 99SE 电子电路绘图软件绘图，图形应当清晰、工整，符合电路制作原则。数字电路绘图原则如下。

① 要标明输入端、输出端以及信号流动方向。

② 通路尽量用线连接，不便连接时，应在端口两端标出，互相连通的交叉线要用小黑点

标出。

③ 同一电路分成两张以上绘制时，应在同一坐标系统，并应标明信号的连接关系。

④ 所有的元器件逻辑符号应符合国家标准。

国内一些电子系统设计工程师实现系统设计的模式是“积木方式”，根据功能框图选择采用合适的通用器件搭建系统。在采购到所需元器件速度、功能完全适用的前提下，“积木方式”不失为一种快捷的设计方法。

复杂的电子系统设计“积木方式”是难以完成的；只有使用顶层设计手段“TOP-DOWN”才能完成；移动电话是“TOP-DOWN”电子系统设计的典型产物。

2．数字系统“自上而下”（Top-Down）的设计方法

数字系统“自上而下”的设计方法的基本思想是：先从顶层进行功能方框图划分和结果分析，实现设计、仿真、测试一体化。具体步骤如下：

（1）系统任务分析与描述手段的确定。描述手段是基础，系统设计师的最初设计思想是从功能描述开始。从系统设计入手，首先要考虑规划出能完成某一具体功能、满足自己产品系统设计要求的某一功能模块，利用某种方式把功能描述出来，通过功能仿真以验证设计思路的正确性。当所设计的功能满足需要时，再考虑以何种方式完成所需要的设计，并能直接使用功能定义的描述。通过系统分析进一步明确待设计系统的逻辑功能。

（2）确定逻辑算法。实现系统算法的方法称为逻辑算法。一个数字系统的逻辑算法往往有多种，设计者的任务不单是找出各种算法，还要确定最合理的一种。

（3）系统划分。当算法明确后，应根据算法构造系统的硬件框架（系统框图），把系统划分成若干个部分。划分后的各部分应逻辑功能清楚，规模大小合适，便于进行电路级设计。

（4）系统（或模块）逻辑描述。当系统中各个子系统和模块的逻辑功能及结构确定后，则需要用较规范的形式来描述系统的逻辑功能。

（5）逻辑电路级设计。选择合理的器件和连接关系以实现系统的逻辑要求。通常采用电路图方式和硬件描述语言（HDL）方式来进行表述。

（6）仿真。现代数字系统设计的 EDA 软件都具有仿真的功能，先通过电路仿真，当验证结果正确后再进行实际电路的测试。

（7）物理实现及其检验。物理实现及检验是指用实际的器件实现数字系统的设计，用电子仪器、仪表测量所设计的电路是否达到设计的要求。

“自上而下”的设计过程并非是一个线性过程，整个设计过程是一个反复修改和补充的过程，是一种由高层次到低层次转换，逐步求精的设计方法。

Top-Down 与传统电原理图设计方法相比有以下优点。

（1）完全符合设计人员的设计思路，从功能描述开始到物理实现的完成。

（2）功能设计可完全独立于物理实现。在采用传统的电原理图输入方法时，不同厂商产品的结构不同，在设计一开始，工程师的设计思路就受到最终所采用器件的约束，大大限制了设计师的思路和器件选择的灵活性。而采用 Top-Down 设计方法可不含有任何器件的物理信息，因此工程师可以有更多的空间去集中精力进行功能描述，设计师可以在设计过程的最后阶段任意选择或更改物理器件。

（3）设计可再利用。设计结果完全可以以一种知识产权（IP，Intellectual Property）的方

式作为设计师或设计单位的设计成果，应用于不同的产品设计中，做到成果的再利用。

（4）易于设计的更改。使用 FPGA 验证算法，设计工程师可在极短的时间内修改设计，对各种 FPGA 结构进行设计结果规模（门消耗）和速度（时序）的比较，选择最优方案。

（5）设计、处理大规模或复杂电路。器件正向高集成度、深亚微米工艺发展。小规模的系统设计，Top-Down 设计方法具有很大的帮助，而大规模的系统设计，Top-Down 设计方法则是必不可少的手段。

（6）设计周期缩短，生产率大大提高，产品上市时间提前，性能明显提高，产品竞争力加强。

对专用集成电路的设计和实现常见有 3 种途径，如图 1-1 所示。

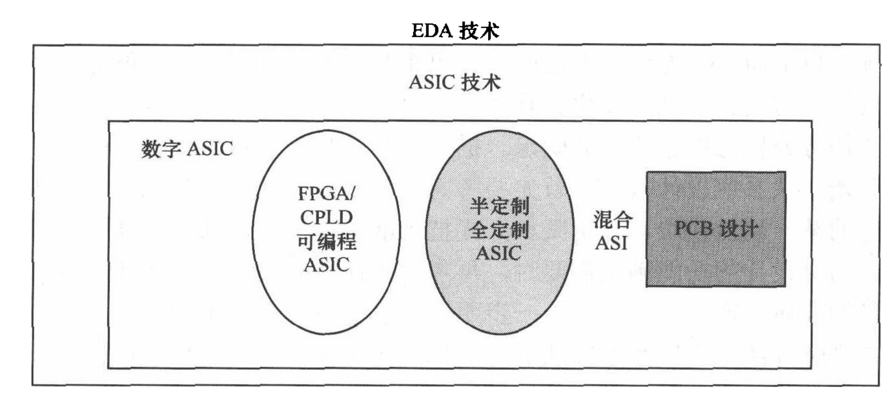


图 1-1　专用集成电路设计和实现的 3 个途径

途径一：使用可编程逻辑器件（FPGA/CPLD）。特点：灵活性，通用性好，上市周期快，对于小批量产品成本低。

途径二：半定制或者全定制 ASIC，包括门阵列 ASIC，标准单元 ASIC 和全定制 ASIC。特点：价格低，性能好，具有知识产权，保密性好。

途径三：混合 ASIC 设计，既具有 FPGA 可编程逻辑资源，又含有可调用的硬件标准单元模块（CPU，RAM，ROM，硬件加法器，乘法器，锁相环）。例如：ALTERA 公司的 Virtex-4 系列，StratixII 系列等。

1.3 体验 EDA

1.3.1 EDA 基本技术特征

总的来说，现代 EDA 技术的基本特征是采用高级语言描述，具有系统级仿真和综合能力。它主要采用并行工程和“自顶向下”的设计方法，使开发者从一开始就要考虑到产品生产周期的诸多方面，包括质量、成本、开发时间及用户的需求等，然后从系统设计入手，在顶层进行功能方框图的划分和结构设计，在方框图一级进行仿真、纠错，并用 VHDL、Verilog-HDL 等硬件描述语言对高层次的系统行为进行描述，在系统一级进行验证，最后再用逻辑综合优化工具生成具体的门级逻辑电路的网表，其对应的物理实现级可以是印刷电路板或专用集成电路。近几年来，随着硬件描述语言等设计数据格式的逐步标准化，不同设计

风格和应用的要求导致各具特色的EDA工具被集成在同一个工作平台上，从而使EDA框架日趋标准化。

1.3.2 EDA的应用范围

EDA技术发展迅猛，应用范围很广，在机械、电子、通信、航空航天、化工、矿产、生物、医学、军事等各个领域都有EDA的应用。目前EDA技术已在产品设计与制造、教学与科研部门广泛使用，并发挥巨大作用。

在教学方面，几乎所有的理工科类（特别是电子信息类）高校都开设了EDA课程。主要是让学生了解EDA的基本概念和基本原理，掌握描述系统逻辑的方法，利用电子器件进行电子电路的模拟仿真实验，并在做毕业设计时选择现代电子电路设计方面的课题。同时，通过学习电路仿真工具，提高学生的实际操作能力，创新能力和计算机应用能力，为从事电子设计方面的工作打下基础。比如参加每两年举办一次的全国大学生电子设计竞赛活动。

在科研方面，主要利用电路仿真工具进行电路设计与仿真；利用虚拟仪器进行产品测试；将CPLD/FPGA器件应用到实际仪器设备中，缩小产品体积，提高产品的性能，技术含量和附加值；进行PCB设计和ASIC设计等。

在产品设计与制造方面，从高性能的微处理器、数字信号处理器，一直到彩电、音响和电子玩具电路等，EDA技术不但用于实现前期的计算机模拟仿真、系统级模拟及测试环境的仿真、产品调试，而且也在电子设备的研制与生产、电路板的设计等方面起着重要作用。

另外，EDA软件的功能日益强大，原来功能比较单一的软件增加了很多新用途，可用于机械和建筑设计，也可扩展到建筑装潢效果图、汽车和飞机的模型、电影特技等领域。

综上所述，EDA技术及其应用软件的广泛使用，极大地方便了设计人员的研发工作，从而为高质量高性能电子产品的开发和生产奠定了基础，可以说EDA技术已经成为电子工业领域不可缺少的技术支持。

1.3.3 EDA的必要性

21世纪是EDA技术的高速发展阶段，EDA技术是电子设计领域的一场革命，随着科技的进步，电子产品的更新日新月异，EDA技术作为电子产品开发研制的源动力，已成为现代电子设计的核心，每年都有新的EDA工具问世。然而，我国EDA技术的应用水平落后于发达国家，因此，作为高等院校相关专业的学生和广大电子工程人员应该尽早掌握这一先进技术，这不仅是提高设计效率的需要，更是我国电子工业在世界市场上生存、竞争与发展的需要。掌握和普及这一全新的技术，将对我国电子技术的发展具有深远的意义。

1.4 本书选用的EDA软件

1. 传统电子设计EDA软件

典型传统电子设计的EDA软件主要实现三项任务：电路原理图的创建、混合信号的仿真和PCB的设计。一般流程是先创建电路原理图，然后进行电路图的仿真，最后在电路原理图基础上设计PCB板。本书选择最具代表性的EDA软件Multisim和Protel，介绍了Multisim

10 和 Protel 99 SE 在电子电路仿真与设计方面的应用。Multisim 软件主要侧重电路的仿真分析，Protel 软件主要侧重电路原理图和 PCB 板的设计。

Multisim10.0 是美国国家仪器（NI）有限公司于 2007 年推出的新版本，它可以实现原理图的捕获、电路分析、交互式仿真、电路板设计、仿真仪器测试、集成测试、射频分析等高级应用。它提供了一个非常大的元器件数据库和齐全的虚拟仪器，具有强大的仿真能力和完美的兼容能力。通过 Multisim10 和虚拟仪器技术可以完成从理论到原理图捕获与仿真，再到原型设计和测试这样一个完整的综合设计流程。

Protel 99SE 是由澳大利亚 Protel Technology 公司推出的基于 Windows 平台下的 EDA 电子辅助设计软件，该软件集成了一系列电路设计工具，如原理图设计工具、PCB 设计工具及自动布线工具等，同时引入了全新的文件管理方式和网络设计机制，可以真正实现电路的高效并行设计。

2. 现代电子设计 EDA 软件

（1）现代电子系统。

数字化是现代电子系统设计的根本标志，集成电路（Integrated Circuit，IC）、专用集成电路（Application Specific Integrated Circuit，ASIC）、片上系统（System on a Chip，SOC）、封装内的系统（System In Package，SIP）是现代电子系统设计发展的历程标志。

“IC”是集成电路的最早定义，出现于 20 世纪 70 年代，80 年代出现了“ASIC”，90 年代“SOC”呈爆炸性发展，至本世纪初，“SIP”、“MCP”（Multi Chip Package）出现在越来越多的场合。

“IC”阶段设计功能级电路，使用分离部件“拼凑”系统各部分功能；“ASIC”阶段设计专业级电路，使用专用芯片设计系统；“SOC”阶段设计系统级电路，直接使用系统芯片开发产品；“SIP”阶段设计产品级电路，针对产品开发产品芯片。

（2）现代电子系统设计 EDA 软件。

本书介绍了 Altera 公司的可编程逻辑器件开发平台 QuartusII8.0 在数字电路仿真与设计方面的应用，同时简单介绍了集成电路制造工艺与专用集成电路设计用到的 EDA 开发软件，如 VCS、Apollo II、Cadence 以及 Hspice 等 EDA 软件。

Quartus II：Quartus II 是 Altera 公司的综合性 PLD 开发软件，支持原理图、VHDL、Verilog HDL 以及 AHDL（Altera Hardware Description Language）等多种设计输入形式，内嵌综合器以及仿真器，可以完成从设计输入到硬件配置的完整可编程逻辑器件（PLD）设计流程。

Quartus II 支持 Altera 的 IP 核，包含了 LPM/Mega Function 宏功能模块库，使用户可以充分利用成熟的模块，简化了设计的复杂性、加快了设计速度。对第三方 EDA 工具的良好支持也使用户可以在设计流程的各个阶段使用熟悉的第三方 EDA 工具。

此外，Quartus II 通过和 DSP Builder 工具与 Matlab/Simulink 相结合，可以方便地实现各种 DSP 应用系统；支持 Altera 的片上可编程系统（SoPC）开发，集系统级设计、嵌入式软件开发、可编程逻辑设计于一体，是一种综合性的开发平台。

VCS：VCS 是编译型 Verilog 模拟器，它完全支持 OVI 标准的 Verilog HDL 语言、PLI 和 SDF。VCS 具有目前行业中最高的模拟性能，其出色的内存管理能力足以支持千万门级的 ASIC 设计，而其模拟精度也完全满足深亚微米 ASIC Sign-Off 的要求。

Apollo II：Apollo II 是世界领先的 VDSM 布局布线工具。它能对芯片集成系统的 VDSM 设计进行时序、面积、噪声和功耗的优化。

Cadence：Cadence Allegro 系统互连平台能够跨集成电路、封装和 PCB 协同设计高性能互连。应用平台的协同设计方法，工程师可以迅速优化 I/O 缓冲器之间和跨集成电路、封装和 PCB 的系统互联。该方法能避免硬件返工并降低硬件成本和缩短设计周期。约束驱动的 Allegro 流程包括高级功能用于设计捕捉、信号完整性和物理实现。由于它还得到 Cadence Encounter 与 Virtuoso 平台的支持，Allegro 协同设计方法使得高效的设计链协同成为现实。

Hspice：Hspice 是 Meta-Software 公司为集成电路设计中的稳态分析，瞬态分析和频域分析等电路性能的模拟分析而开发的一个商业化通用电路模拟程序，目前已被许多公司、大学和研究开发机构广泛应用。Hspice 可与许多主要的 EDA 设计工具，诸如 Candence，Workview 等兼容，能提供许多重要的针对集成电路性能的电路仿真和设计结果。采用 Hspice 软件可以在直流到高于 100MHz 的微波频率范围内对电路作精确的仿真、分析和优化。在实际应用中，Hspice 能提供关键性的电路模拟和设计方案，并且应用 Hspice 进行电路模拟时，其电路规模仅取决于用户计算机的实际存储器容量。

习 题 一

1．简述 EDA 技术的概念。
2．EDA 技术的发展经过了哪几个阶段？
3．简述 EDA 技术的基本特征及其应用范围。
4．简述 Multisim10 和 Protel 99 SE 两款 EDA 软件。
5．简述 EDA 技术的发展趋势。

第2章 计算机辅助电路分析和电路仿真

本章要点

- Multisim10 电路原理图的创建
- Multisim10 虚拟仪表的使用
- Multisim10 的分析方法

电子系统分为模拟电路和数字电路两种。由模拟器件所组成的模拟电路，一般研究的是电路中各处电流、电压等模拟变化量之间的关系，其输入输出大多数是波形或数值；而由各式各样的逻辑门和功能块组成的数字电路，则主要研究的是信号彼此之间的逻辑关系，其输入输出是0、1和x（不确定）等电平。因此，用计算机仿真分析这两种电路的方法有些不同。

2.1 模拟电路仿真原理

模拟电路的仿真分析是以电路理论、数值计算方法和计算机技术为基础实现的。它借助于计算机的计算、存储和图形处理的高速和高效率，采用特定的数学模型和仿真算法，用预先设计出的各种仿真分析的应用程序对电路进行各种分析、计算和验证。模拟电路的仿真过程如图2-1表示。

从图2-1中可以看出，模拟电路的仿真过程由以下几部分组成：输入部分（输入电路结构参数及仿真要求）、元器件模型库（创建和存放元器件数学模型）、通用仿真程序（自动建立相关电路方程并进行数值求解）和输出部分（处理和输出仿真结果）。

2.1.1 输入方式

模拟电路的输入方式有两种：电路网表文件输入和电原理图输入。用电路网表输入是一件既繁琐又容易出错的工作，对规模较大的电路，这一问题尤其突出。随着计算机图形处理技术的发展，现在的电路仿真工具基本上都采用方便直观的电原理图输入方式。

采用电原理图输入方式的仿真工具要建立一个电路符号库。建立电路符号库的方法很多，如源文件编译法、交互式图形编辑法等。不管用哪种方法建库，库中任一个电路图形符号一般都可以通过6个方面的信息来描述：图形符号名、子电路路径、项目种类代号、符号尺寸

及类型、符号的引线描述及符号现状描述。其中不同元器件图形符号描述又可分为3类：块状符号、点阵图符号和矢量数据格式符号。

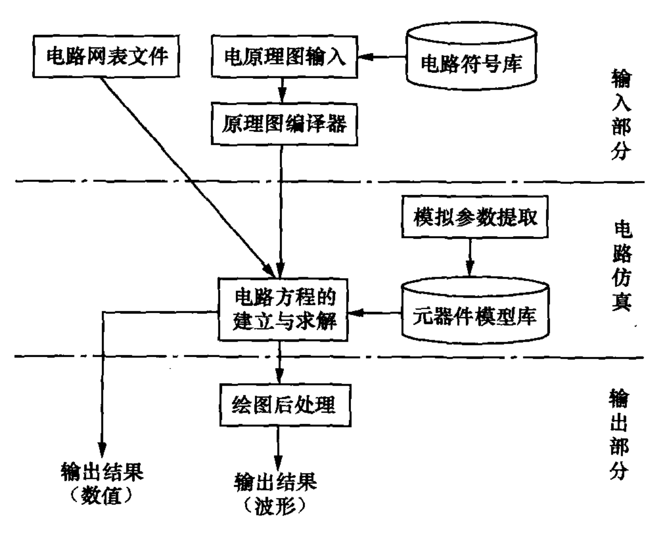


图2-1 参数扫描分析结果

从电路符号库中调出所需电路图形符号，组成电路原理图，由原理图编译器自动将原理图转化为电路网表文件，并自动标上节点号，提供给仿真工具进行仿真。

2.1.2 元器件模型

与实际电路一样，电路仿真中也离不开各式各样的元器件，只不过仿真中所用到的元器件都是实际元器件所对应的数学模型。这些元器件模型要求能够准确地反映实际元器件的物理特性和电学特性，以便于计算机做数值计算。由于模拟器件的种类繁多，元器件的制造工艺、工作条件、非线性特性等都对模型的建立有较大影响，因此，建立一个合适的元器件模型是一项难度较大的工作。

建立元器件模型的方法大体上有两种：一种是从分析元器件物理特性和电学特性的基础上来构造模型，这要求元器件本身的物理机理比较清楚，其物理特性和电学特性不太复杂，如电阻、电容、电感、电压源、晶体管、场效应晶体管等；另一种是根据元器件的输入/输出特性参数来构造模型，即所谓的“宏模型”，如运算放大器、电压比较器等特性较为复杂的元器件。

元器件模型的精确程度直接影响着电路仿真结果的准确性。为了满足不同的仿真精度要求，有一些元器件需要提供不同层次、不同类型的仿真模型，如双极型晶体管就有Ebers-Moll（EM）和Gummel-Poon（GP）两种模型。EM又在不同的工作条件下细分为EM1、EM2和EM3，其中EM1仅考虑了管子的直流特性，EM2增加了交流特性，而EM3则充分考虑了电荷存储、β随电流变化、基区宽度调制等效应。GP是一种数学推导更加严密和完整的模型，包括了器件主要的二次效应。模型的精度越高，仿真结果就越准确，但仿真的计算量和存储量就会越大。同时，元器件模型库中所装有的元器件模型的数量又决定了仿真工具的有效仿真范围。我们经常见到的仿真软件有专业版、教育版、学生版等版本之分，其中一个主要区别就在于元器件模型库数量的多少。元器件模型的精度和元器件模型库中仿真模型的数目也是评价一个电路仿真软件性能高低的一个非常重要的指标。

2.1.3 电路方程的建立与求解

电路方程是电路仿真分析的核心，电路仿真程序根据输入电路结构、元器件参数和仿真要求，能自动建立起相应的电路方程。常用的建立电路方程的方法有：节点法、改进节点法、状态法、混合法等，其中又以改进节点法为大多数仿真软件所采用，如 Spice、Multisim 等软件。所谓改进节点法，就是当电路两节点间仅存在一个理想的电压源或受控电压源，用一般节点法列写节点方程时将出现无穷大的导纳，这时在理想电压源支路引入电流变量 I，作为附加的待求变量，列写到节点方程中，这就是改进节点法的基本思想。

对于不同的电路结构和不同的电路分析方法，所列出的电路方程是不一样的。如求线性电路的直流工作点，列出的电路方程肯定是线性代数方程；而含有 RLC 动态元件的瞬态分析则对应于微分方程；非线性电路所对应的电路方程一般是非线性方程。

电路仿真中求解电路方程多采用数值分析方法（数值解法）。根据电路方程的不同，又有好多种算法，如线性代数方程有高斯消元法或 LU 分解法，非线性代数方程有牛顿迭代法，而微分方程则采用显式积分法或隐式积分法等。求解电路方程经常要进行矩阵运算，电路的节点越多，矩阵就越大，运算就越费时；电路节点越多，0 元素所占比例越高，其矩阵就越稀疏，这就是所谓的稀疏矩阵。因此在进行数值计算时，常用到稀疏矩阵技术，以提高仿真效率。另外，在非线性代数方程和微分方程的求解过程中，还需解决稳定性、收敛性等问题。

电路方程的建立与求解的过程虽然复杂，但这些方法由仿真软件完成，不需要设计者计算。

2.1.4 图形的后处理

经过电路仿真计算后得到的计算结果通常是数值数据，这和我们在实际电路测试中见到的曲线、波形不一样，不符合我们的阅读习惯。图形的后处理程序就是将这些数据转换成人们所习惯见到的波形和曲线。处理过程中可以一边仿真、一边实时地显示结果；也可以在仿真结束时做统一显示或运算。为了更形象地模仿实际的测量工作，有些仿真工具把图形的后处理工作做成与实际仪器（仪表）相似的图形形状，如 Multisim 仿真软件中的示波器、信号发生器、波特仪等，使它们类似于虚拟仪表。

2.2 数字电路的模拟

用计算机根据给定的数字电路拓扑关系以及电路内部数字元器件的功能和延迟特性，分析计算整个数字电路的功能和特性，这就是数字电路的逻辑仿真，习惯上称为逻辑模拟，所以把数字电路又称为逻辑电路。显然这里所提及的“模拟”与本章 2.1 节“模拟电路”中的“模拟”是完全不同的两个概念。逻辑模拟的对象一般仅局限于由门电路和逻辑功能块等元器件组成的逻辑电路，所以也称为门级电路和功能块级电路的模拟。

数字电路模拟的目的就是检查数字电路是否具有设计要求的功能，包括逻辑功能和延迟特性、负载特性等。模拟的方法一般是在电路的外部输入端上加激励信号波形，通过信号沿着元器件和线路向输出端传输，在输出端上得到响应波形。通过观察和分析波形的时序关系判断其功能是否正确。

2.2.1 数字电路模拟的过程

数字电路模拟的基本过程如图 2-2 所示。从图中可以看出，数字电路的模拟过程分为两个阶段："准备工作"和"运行模拟"。由于数字电路是用逻辑图或硬件描述语言来表征，因此在运行模拟之前，需要对它们进行转换或编译处理，生成便于计算机分析运算的数字电路"网表"文件并存入到数据库中。在数据库中除了网表数据外还需存有大量的供逻辑模拟的元件模型数据。对于逻辑模拟中所用到的激励信号则用波形描述语言（或图形）建立输入波形文件，存放到当前工程（当前工程文件夹）中。

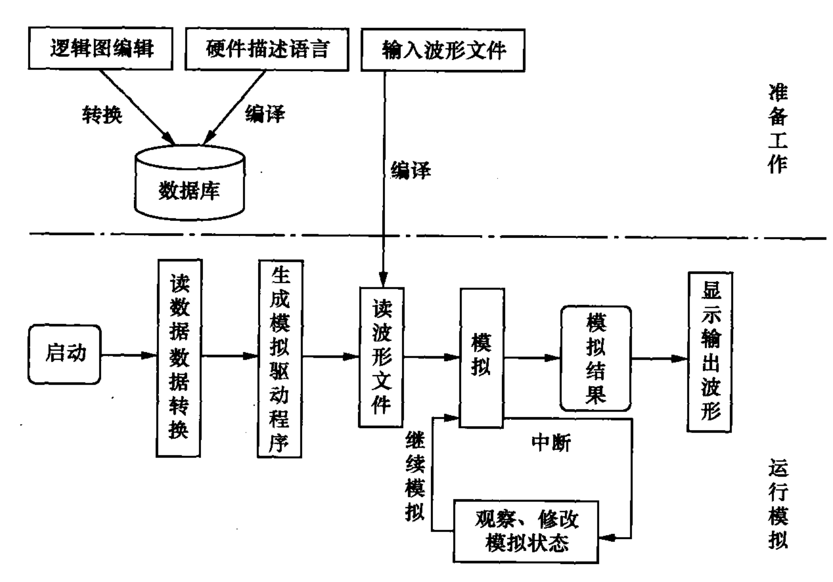


图 2-2　数字电路的逻辑模拟过程

运行模拟时，首先从数据库中读入网表数据和模型数据及波形数据，经检查正确无误后转换成模拟用内部数据。

在包含功能块的模拟系统中，因为根据用户描述所生成的功能计算子程序一般是以函数库的形式保存的，所以要将模拟程序与功能计算子程序连接在一起，形成可执行的模拟程序，然后自动运行该模拟程序。

所生成的模拟程序启动运行之后，首先读入激励波形和模拟控制命令，接着就在这些激励波形和模拟控制命令的控制下进行模拟。

模拟结果包括输出波形文件和电路错误报告。波形文件随着模拟时钟的前进不断添加新的模拟结果，可由波形编辑器同步地动态显示。波形编辑器还可以独立地显示的波形进行编辑，如上下左右滚动、信号调换位置、信号分组等。电路错误报告保存模拟过程中得到的检查结果，如语法错误、竞争冒险情况、时序检查结果等，为设计者修改设计做必要的参考。

当模拟到达规定时间或模拟过程中已经不存在任何触发事件时，逻辑模拟过程才结束。模拟过程进行中也可以由用户中断。中断的手段有命令方式和随机方式两种，命令方式在输入波形文件时指定断点，包括时间断点和条件断点；用户也可以在模拟过程中随时发出中断命令，在中断之后用户可以观看波形，查看错误报告。

2.2.2 逻辑模拟的模型

为了进行逻辑模拟，必须建立电路的逻辑模型。逻辑模型是对所模拟（即仿真）电路中元件的逻辑行为进行某种量化的表示，它主要用来反映实际电路的特性。逻辑模型越接近实际电路，模拟的结果越精确，但往往也增加了复杂度。选取什么样的逻辑模拟，取决于模拟的目的和对模拟精度的要求。

1．逻辑信号模拟

在数字电路中，信号值常用 0 和 1 两值表示。但有时 0、1 不能完全表征实际电路的某些状态，如当信号由 0 变为 1，或由 1 变为 0 的过渡期间，信号既不是 0 也不是 1，而是一种不确定的状态；再如当晶体管截止时常出现高阻状态，它相当于把线路断开。这些状态都不宜用 0、1 来表征，所以在逻辑模拟中除了 0 与 1 两个确定的逻辑信号值，有时还需要人为地引入其他的逻辑值。这样，信号的逻辑关系就不再是二值逻辑，而是多值逻辑。除了信号值本身外，有时还需要区分相同信号值的不同驱动能力，称为信号值强度或电平强度。例如，在 MOS 电路中常用不同的电阻值来控制其工作过程。选取不同信号值的组合就构成了各种功能的逻辑信号模型。

（1）三值逻辑。三值逻辑就是除了 0、1 外，增加了一个未知信号值 X。三值信号模型常用来检测逻辑电路的竞争与险象。

（2）四值逻辑。这是三值逻辑的扩充，它在 0、1、X 之外，又引入一个称之为高阻值的 Z，用来表示一个信号与其源断开后的状态。如一个单向开关的输出端，在开关导通时有一个从输入端传过来的确定的逻辑值，但在开关断开后，该输出端既有可能保持原信号值，但又无能力长久维持原信号值不变，这一特殊状态值便用高阻值 Z 表示。四值逻辑常用来实现对 MOS 电路的模拟。

（3）五值逻辑。在三值逻辑中，用 X 表示跳变时不能区分上跳和下跳。为了明确表示跳变的不同情况，引入 U 和 D 分别表示上跳和下跳，而用 E 替代 X 来表示未定状态。这里由 0、1、U、D、E 组成了所谓的五值信号模型。

除三值、四值和五值信号模型外，还可以根据特殊要求设定其他的值，如六值逻辑、八值逻辑和 IEEE 标准的九值逻辑等模型。随着值的增加无疑会提高模拟能力和精度，但也会相应增加模拟过程的运行时间。

（4）信号值强度。在逻辑模拟中，除了考虑逻辑信号值外，还有考虑到另外一种现象，如直接连接到电源的信号和通过一个较大的电阻与电源相连接的信号，尽管它们都可能有相同的信号值（如逻辑 1），但很明显它们的驱动能力是不一样的。为了区分这类现象，就引入了信号值强度这一概念，因此当不同的逻辑信号作用于同一节点时，要按逻辑信号值及其强度才能完全确定节点的状态。信号值强度按强度的强弱顺序一般分为如下三级。

① 强制级（F）。表示该信号连接电源或地，或者是一些输入激励。在电路中信号通过处于导通状态下的晶体管与电源或地相连接，如果其电阻很小，如 CMOS 门的输出，也认为是强制级。

② 电阻级（R）。当信号通过一个较大的电阻与电源或地相连接，则认为是电阻级，如 NMOS 电路中当晶体管截止时信号通过电阻较大的上拉晶体管与电源相连，就可以认为是电阻级，再如集电极开路的输出也属于电阻级强度。

③ 高阻级（Z）。由于晶体管的截止，信号与电源和地相隔离，则可以认为是高阻级，

如三态门控制端信号无效时的输出。

2．逻辑电路网表

模拟电路中的输入方式可采用“网表”的形式输入，其实采用电原理图输入，经过原理图编译器实际上也就是转换成“网表”形式。对于需要模拟的数字电路来说，其电路信息也是以“网表”的形式存放到数据库中再进行调用的。数字电路的网表和模拟电路的网表相似，两者都以文本形式描述，都可能用来显示需要仿真（模拟）电路的信息，但两者书写格式不一样。

3．基本逻辑元件模型

数字电路中最基本的元件是门，完全由门组成的数字电路称为门级电路，只处理门级电路的模拟称为门级模拟。门级模拟通常也可以处理常见的各种类型的触发器。为了提高模拟效率，常把电路中具有一定功能的部分电路作为一个模拟元件，这样的元件称为功能块。功能块的模型用功能和行为来描述，而不关心其内部结构和组成，含有功能块元件的模拟称为功能块级模拟。通常把含有门和功能块的模拟称为逻辑模拟。

逻辑元件的模拟需给出各类元件的内部功能、参数及特性，如输入输出端个数、延迟时间、扇入扇出系数等。

（1）基本逻辑门模型。基本逻辑门有以下 8 种：与门（AND）、与非门（NAND）、或门（OR）、非门（NOT）、或非门（NOR）、异或门（XOR）、异或非门（NXOR）、缓冲器（BUF）。非门 NOT 是单输入单输出；缓冲器（BUF）是单输入双输出，一个输出与输入信号相同，另一个输出则与输入信号相反；其余的基本逻辑门都是多输入单输出。

逻辑门的延迟时间是指信号从输入端传播到输出端所需要的时间，按照所规定的延迟模型决定其值。扇入系数和扇出系数反映元件带负载的能力，扇入系数反映输入端的负载电流，扇出系数反映输出端的驱动电流。

（2）三态门模型。三态门是带使能控制端 E 的逻辑门，它是 MOS 电路中最常用的一种门，由基本门加一个控制端构成。如三态反相门，当 E=1 时，它的功能和一般反相门功能相同，而当 E=0 时其输出呈高阻态。

由于输入端到输出端的延迟与由使能控制端 E 到输出端的延迟一般不相等，在模型中要分别指定。

（3）功能块模型。功能块常见有寄存器、存储器、译码器、加法器、比较器等，其模型一般只给出其功能，而不管其内部结构。

功能块通常有多个输入端和输出端，且每个输入端和输出端一般是不对称的，它们的扇入/扇出系数、延迟时间等往往也各不相同。在功能描述中一般分别描述各个端口的特性，包括信号值的计算和延迟时间的计算。有的功能块具有记忆功能（时序逻辑），即其内部含有记忆元件。除了输入端和输出端外，还有用来记忆内部状态的内部信号。

4．逻辑信号的延迟模型

把实际电压或电流传播的滞后效应，按一定规则简化为用一个或几个参数来表征逻辑信号的延迟，称为延迟模型。延迟模型越逼近信号滞后的真实情况，模拟结果的可信度越高，但模拟过程复杂且费时。简单的延迟模型，可简化模拟过程，节省模拟时间，但要牺牲或放

宽对模拟结果真实性的要求。元件的延迟时间一般定义为信号从元件的输入端到该元件的输出端所需要的时间。常见的延迟模型有以下几种：

（1）零延迟模型。所有的元件其传输延迟均设置为 0。这虽然不符合实际情况，但用这种模型可以验证组合逻辑电路或同步时序逻辑电路的逻辑功能的正确性。不过这种模型不便于模拟异步时序电路。

（2）单位延迟模型。电路中所有元件都赋予相同的值，并取值为 1。这也与实际情况不符，所以它也只能用于验证逻辑功能的正确性，可以用于异步时序电路的逻辑验证。

（3）标准延迟模型。根据元件特性，对每种元件规定一个标准延迟数值，通常按产品目录中给定的数据为依据。它不考虑同类元件的参数分散性，与真实情况仍有差别，但由于在逻辑设计阶段还不能确定参数分散情况，因此标准延迟模型对于大多数电路已经足够精确。

（4）模糊延迟模型。为了反映参数的分散性，延迟时间不是给定的一个值，而是给定的一个范围，即给出延迟最小值和最大值。在这个范围内，其信号值不定（用 X 表示），即有一个模糊区域。

为了提高逻辑模拟精度，根据实际模拟的需要，还可以定义若干种延迟模型，如上升/下降延迟模型、元件特性延迟模型、连线延迟模型、惯性延迟模型等。

2.2.3 逻辑模拟的算法

逻辑模拟算法的基本思想是，将施加在电路外部输入端的激励信号通过电路中各元件向输出端传播，从而得到各节点的信号值。

从原理上讲，任何数字电路的逻辑关系都可用如下方式表示。

对于组合电路： $y_t=B(x_t)$

对于时序电路： $y_t=B(x_t,\ Q_t)$ $Q_{(t+\Delta t)}=B_2(x_t,\ Q_t)$

其中，x 为输入矢量，y 为输出矢量，Q 为内部状态矢量，B 为布尔函数，t 为时间。若要考虑时间延迟，还应在上述表达式中附加上延迟算子。如果直接采用以上两式进行计算机逻辑模拟，其效率较低。在开发实用的逻辑模拟软件的过程中，通常需要考虑如何采用快速、有效、可靠的算法。以下是几种典型的逻辑模拟算法。

1. 编排级数法

编排级数法是一种早期使用的逻辑模拟方法。该算法首先针对各类逻辑元件编制各自的求值子程序。逻辑模拟主程序则根据网络中各元件的位置，依一定顺序就每个元件调用一次与之相对应的求值子程序，对每个元件进行顺序求值。为了安排元件的求值顺序，则需对每个元件设置一个级数。把电路的输入端口定为 0 级，某个元件的级数等于其所有前缀元件的最大级数加 1。在逻辑模拟主程序中，就按各元件的级数从小到大的顺序调用与各元件相对应的求值子程序。这种算法实际是默认零迟延，只适合做功能验证。

2. 事件驱动法

当数字电路的输入发生变化时，通常系统中只会有少数节点的逻辑信号值发生改变，我们把节点信号的每一次变化称为一个“事件”(event)。事件驱动算法的基本思想就是在逻辑模拟过程中，只对输入信号有变化的元件，或输出信号可能有变化的元件进行求值。

在外部激励信号的作用下，逐渐得到各级元件的响应，由事件驱动信号的负载元件并根据元件的功能计算元件输出端的逻辑值，得到新的事件，这样逐级传播，直到不再存在新的事件，或者到达指定的结束时刻为止。

3. 时钟驱动法

时钟驱动法是针对超大规模数字集成电路所采用的一种逻辑模拟算法，主要为了解决逻辑模拟时的快速收敛问题。其考虑问题的出发点是：将超大规模集成电路看成一个同步时序逻辑电路，电路中有一个或几个时钟控制着各子系统的运行。在时钟触发电路时，系统同步地从一种稳定状态转至另一种稳定状态，对系统性能进行宏观分析时，可忽略状态过渡的微观特性。即一个复杂的数字电路（数字系统）可看作仅是系统时钟的函数，从而大大减少了计算量和存储量。

2.3 数模混合仿真技术

我们已经知道，模拟电路的仿真分析和数字电路的逻辑模拟在计算方法和输出结果类型等方面存在着较大差异。前者是用数值计算方法求解电路方程组，计算的是节点电位和支路电流；而后者则采用事件驱动等算法分析计算电路节点的逻辑状态（如“1”、“0”和“X”等)。因此，早期的电路模拟和逻辑模拟一直在各自不同的仿真软件中进行。但随着电子技术的发展，在电路设计中同时包括有数字和模拟单元的情况越来越多，这就要求仿真工具同时具有仿真模拟电路和数字电路以及它们的混合电路的功能，这就是所谓的数模混合仿真技术。现在的电路仿真工具大多都具有数模混合仿真能力。解决数模混合仿真问题的主要技术有：“顺序模拟”和“混合模拟”。

2.3.1 顺序模拟

当逻辑和模拟单元之间无反馈连接关系，即可以将整个仿真系统分析分成模拟和逻辑单元时，利用顺序仿真方法实现数模混合电路仿真。数模混合电路中根据连接到各节点上的电路元件类型不同，存在着 4 种类型的电路节点。

（1）数字节点：连接至数字节点的所有电路元件均是逻辑器件。

（2）模拟节点：连接至模拟节点的所有电路元件均是模拟器件。

（3）A/D 节点：指从模拟元件转至逻辑元件的节点。

（4）D/A 节点：指从逻辑元件转至模拟单元的节点。

在数模混合电路中，关键问题是如何处理 A/D 节点和 D/A 节点。A/D 节点和 D/A 节点也称为接口。对于 A/D 节点，模拟电压通过 A/D 转换器接口转换为逻辑电平，即模拟电压值通过与阈值进行比较，归为可能的逻辑值 0，1，X 等。对于 D/A 节点，逻辑信号通过 D/A 转换器转换为模拟电压，既有限的逻辑电平转换为无限的模拟值。

实际电路中 A/D、D/A 节点在电路中出现的情况如图 2-3 所示，有以下几种情况：

（1）只包含 A/D 节点系统的仿真；

（2）只包含 D/A 节点系统的仿真；

（3）A/D—D/A 或 D/A—A/D 多级数模混合系统的仿真。

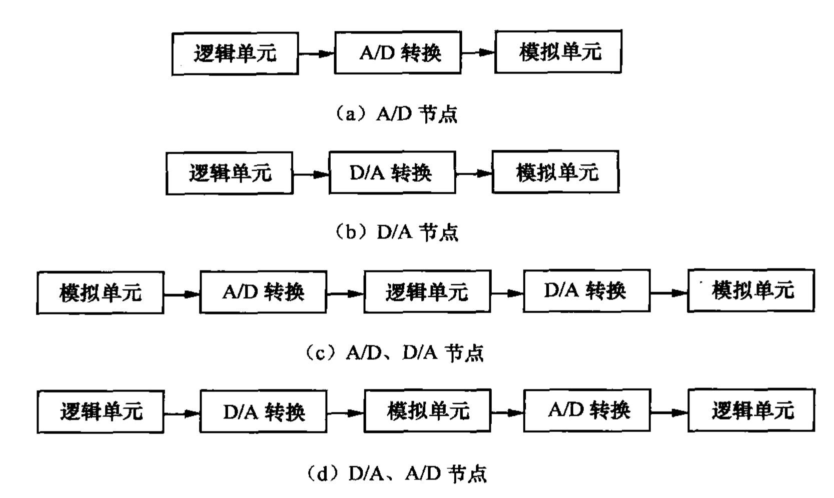


图 2-3 电路图中出现 A/D、D/A 节点的几种情况

顺序仿真时，首先从 A/D 节点和 D/A 节点处将电路分成模拟单元和逻辑单元，然后利用单独的程序对模拟单元和逻辑单元分别仿真，各部分的仿真再通过 A/D 或 D/A 接口进行耦合、连接。如图 2-3（a）所示的电路连接中，仿真程序首先计算模拟单元的电压值，然后将模拟电压传输给 A/D 接口。A/D 接口将模拟电压转换成逻辑电平。A/D 接口的输出信号作为逻辑单元的输入激励信号，利用逻辑仿真程序进行逻辑单元的仿真，从而完成数模混合电路的仿真工作。

对于图 2-3（c）所示的电路，同样可以应用顺序仿真的方法，只是进行逻辑仿真和模拟仿真的次数较多。在顺序仿真中，模拟仿真和逻辑仿真需要单独的程序。

2.3.2 混合模拟

若数模混合电路中不同部分之间存在反馈，很难明确地将电路分成数字、模拟几个部分，则不能采用顺序仿真的方法，需采用直接进行混合仿真的方法。该方法的基本要点如下。

（1）电路中数字逻辑单元器件及加于数字单元输入端的激励信号按规定格式在输入文件中描述，对电路中的接口节点，程序将按其是 A/D 节点还是 D/A 节点，分别自动插入数字输入元件和数字输出元件。

（2）按下述处理方法对整个电路进行数模混合模拟。

① 进行直流（DC）分析时，将所有逻辑单元器件的延迟时间设置为零。将数字信号源的值设置为其 t=0 时的初始值。

② 逻辑单元器件不参与交流小信号 AC 分析。在进行 AC 分析时所有逻辑单元器件保持其在直流工作点计算时的逻辑状态。

③ 进行瞬态分析时首先按照原则①进行直流工作点分析，确定分析瞬态的初值，然后采用对纯模拟电路进行瞬态分析的同样方式进行数模混合电路的瞬态分析。但数字部分和模拟部分的瞬态分析步长并不相同，由程序进行内部动态调节。对逻辑单元器件，若延迟时间小于时间步长的一半，则认为延迟为零。对脉宽小于延迟时间的输入脉冲将不予考虑。

2.4 常用的电路仿真工具

电子电路设计与仿真工具包括 SPICE/PSPICE、Multisim、Matlab、SystemView、MMICAD LiveWire、Edison、Tina Pro Bright Spark 等。下面简单介绍前 3 个软件。

（1）SPICE（Simulation Program with Integrated Circuit Emphasis）：由美国加州大学推出的电路分析仿真软件，是 20 世纪 80 年代世界上应用最广的电路设计软件，1998 年被定为美国国家标准。1984 年，美国 MicroSim 公司推出了基于 SPICE 的微机版 PSPICE（Personal-SPICE）。1985 年，加州大学伯克利分校用 C 语言对 SPICE 软件进行了改写，并由 MICROSIM 公司推出。1988 年 SPICE 被定为美国国家工业标准。与此同时，各种以 SPICE 为核心的商用模拟电路仿真软件，在 SPICE 的基础上做了大量实用化工作，从而使 SPICE 成为最为流行的电子电路仿真软件。PSPICE 采用自由格式语言的 5.0 版本，自 20 世纪 80 年代以来在我国得到广泛应用，并且从 6.0 版本开始引入图形界面。PSPICE 软件具有强大的电路图绘制功能、电路模拟仿真功能、图形后处理功能和元器件符号制作功能，以图形方式输入，自动进行电路检查，生成图表，模拟和计算电路。它的用途非常广泛，不仅可以用于电路分析和优化设计，还可用于电子线路、电路和信号与系统等课程的计算机辅助教学。与印制版设计软件配合使用，还可实现电子设计自动化。被公认为通用电路模拟程序中最优秀的软件，具有广阔的应用前景。

（2）Multisim（EWB 的新版本）软件：Interactive Image Technologies. Ltd 在 20 世纪末推出的电路仿真软件，目前普遍使用的是 Multisim10，相对于其他 EDA 软件，它具有更加形象直观的人机交互界面，特别是其仪器仪表库中的各仪器仪表与操作真实实验中的实际仪器仪表完全没有两样，它对模数电路的混合仿真功能毫不逊色，并且它在仪器仪表库中还提供了万用表、信号发生器、瓦特表、双踪示波器（对于 Multisim10 还具有四踪示波器）、波特仪（相当实际中的扫频仪）、字信号发生器、逻辑分析仪、逻辑转换仪、失真度分析仪、频谱分析仪、网络分析仪、电压表、电流表等仪器仪表。还提供了我们日常常见的各种建模精确的元器件，如电阻、电容、电感、三极管、二极管、继电器、可控硅、数码管等。模拟集成电路方面有各种运算放大器、其他常用集成电路。数字电路方面有 74 系列集成电路、4000 系列集成电路等，并支持自制元器件。Multisim10 还具有 IV 分析仪（相当于真实环境中的晶体管特性图示仪）、Agilent 信号发生器、Agilent 万用表、Agilent 示波器等。

（3）Matlab 产品族：它们的一大特性是有众多的面向具体应用的工具箱和仿真块，包含了完整的函数集用来对图像信号处理、控制系统设计、神经网络等特殊应用进行分析和设计。它具有数据采集、报告生成和 MATLAB 语言编程产生独立 C/C++代码等功能。Matlab 产品族具有下列功能：数据分析，数值和符号计算，工程与科学绘图，控制系统设计，数字图像信号处理，财务工程，建模、仿真、原型开发，应用开发，图形用户界面设计等。Matlab 产品族被广泛应用于信号与图像处理、控制系统设计、通信系统仿真等诸多领域。

2.5 Multisim 10 的基本操作

本章只介绍 Multisim10 的最基本和最常用的功能。掌握了这些功能，就能够熟练地将该

软件的仿真能力应用到电工基础、电子技术、自动控制等课程中，同时也可以帮助电子工程师解决实际问题。

2.5.1 电原理图的创建

1．Multisim 界面简介

设计电路原理图是进行电子产品设计的重要环节，本小节将介绍设计电路原理图的各项基本操作。首先来介绍一下 Multisim10 的界面。

从“开始”菜单启动 Multisim10 软件后，屏幕出现 Multisim10 软件的主窗口，如图 2-4 所示。主窗口包括主菜单、工具栏、设计工具栏、仪器仪表工具栏和设计窗口。

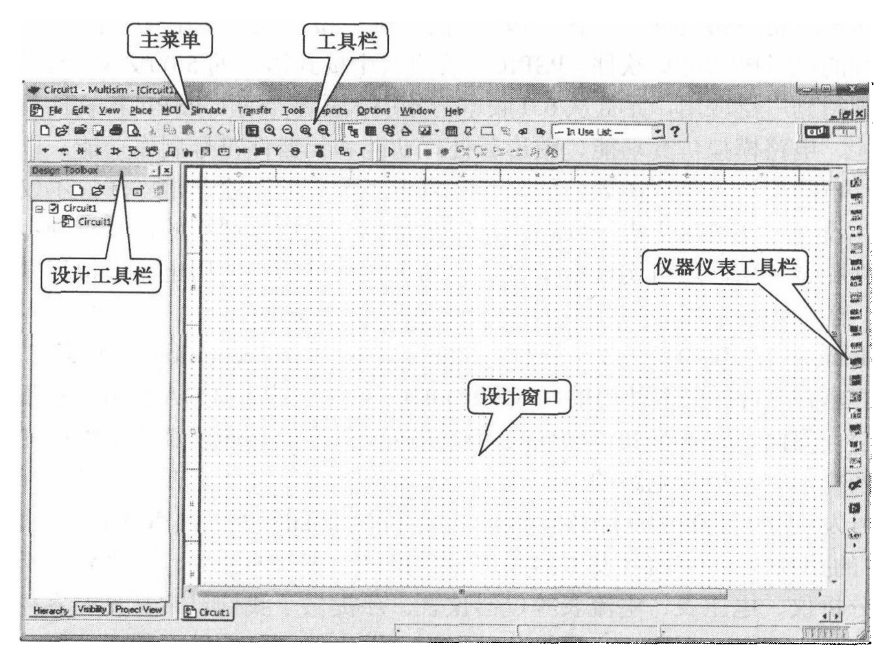


图 2-4 Multisim10 软件的主窗口

主菜单包括 File、Edit、View 等菜单，主要是用于文件的创建、管理、编辑、窗体的管理以及电路仿真软件的各种操作命令。

工具栏提供了常用命令的快捷操作方式，方便用户使用。仪器仪表工具栏提供了在电路仿真过程中会用到的各种虚拟仪器。Multisim10 为设计者提供了大量的虚拟仪表，如数字万用表、失真度分析仪、函数信号源、双踪示波器、扫描源、频谱仪、LabView 虚拟仪器等。利用这些仪器仪表，用户可以对设计的电路进行各种综合分析和测量。

2．原理图的创建

（1）打开 Multisim10 的工作平台。

① 单击“开始/所有程序/National Instruments/Circuit Design Suite 10.0/Multisim”菜单命令。

② 双击 Multism10 应用程序图标 Multisim。

这两种方法均能打开一个空白文档，并命名该电路（文件）的名称默认为“Circuit1”。

（2）更改电路名称。默认电路名称为“Circuit1”，也可以由用户重新命名。在菜单栏中选择“File/Save As”命令，系统弹出标准的 Windows 存储对话框，提示用户此文件存于什么路径，用什么文件名。

（3）打开一个已存在的文件。单击“File/Open”菜单命令，找到已存在的文件存放路径，选中此文件，单击“打开”按钮即可打开文件。如果要打开的文件是早期 Mutisim 版本的文件或其他仿真软件的文件，则在打开文件的对话框中可以看到一个文件类型下拉菜单，选择要打开文件相应的版本，就可以在 Multisim10 中打开这一文件。例如，打开一个文件名为“模数转换及数字显示电路”的文件，出现如图 2-5 所示的结果。

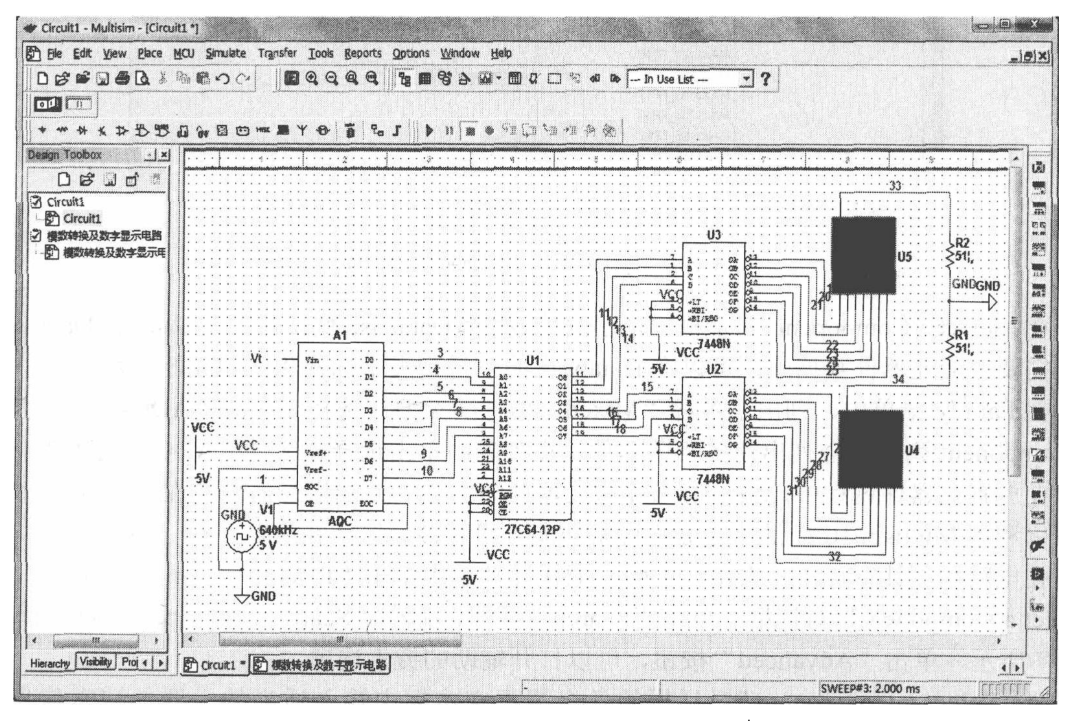


图 2-5　打开“模数转换及数字显示电路”

3．元器件的放置

Multisim10 采用电原理图输入方式，因此电路图的创建非常方便。如果我们要建立如图 2-5 所示的电路文件，首先要选择电路图中所要的元器件。

（1）元器件的选取方法。

① 从器件工具栏的器件库中选取。从器件工具栏的器件库中直接选取元器件是常用的一种方法。选取元器件时，首先要知道该元器件是属于哪个元器件库，然后将光标指向所要选取的元器件所在的元器件分类库，在弹出的“Select a Component”对话框中找到自己所要用的元器件，单击“OK”按钮即可将元器件调出。例如，我们要放置一个“ADC”，首先让光标指向“Mixed”库，单击后即弹出“Select a Component”对话框，如图 2-6 所示。

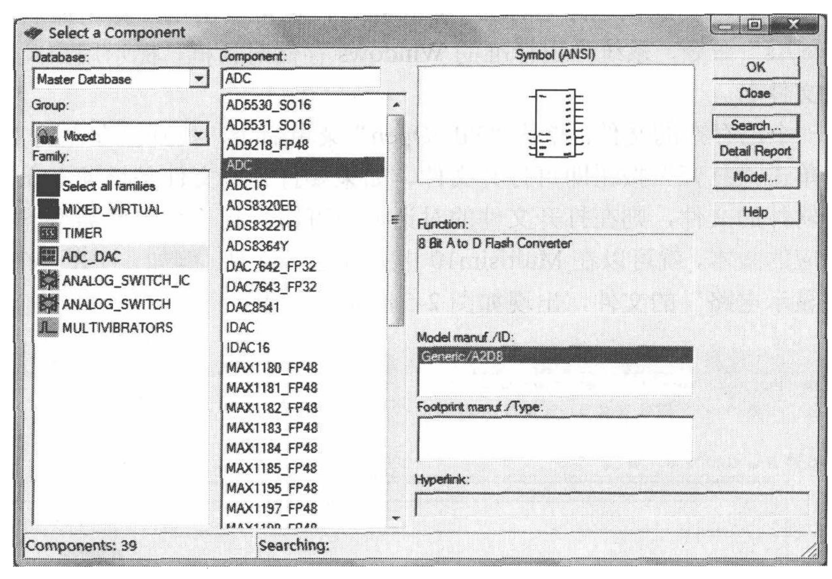


图 2-6 “Select a Component”对话框

在“Select a Component”对话框中，元器件数据库 Database 默认设置为“Master Database”，如果需要改变元器件数据库，可以在 Database 的下拉列表中选择相应的元器件数据库。

在 Family 菜单中，列出了各种“Mixed”器件的类型，选择相应的“ADC-DAC”类型，在 Component 列表中会列出众多相关器件，在列表中单击选择相应的“ADC”元器件。

在放置元器件过程中，很可能只知道元器件的名称，不知道元器件所在的元器件库，这时可以利用 Multisim10 提供的元器件搜索功能查找到该元器件并放置。

在电路图设计窗口下，右击，在弹出的快捷菜单中单击“Place Component”命令，打开“Select a Component”对话框，单击“Search”按钮，弹出“Search Component”对话框，如图 2-7 所示，单击“Advanced”按钮，可以打开辅助的搜索选项。

在“Select a Component”对话框的各个搜索文本框中输入适当的元器件搜索信息（至少要输入其中的一项），即可开始搜索。在搜索过程中，也可以用“*”代替不确定的字符串。

② 使用菜单命令放置元器件。Multisim10 提供有放置元器件的菜单命令 Place/Component，执行该命令，将出现与图 2-6 所示完全一样的“Select a Component”对话框，其操作方法也完全相同。

③ 使用“In Use List”放置相同的元器件。在工具“In Use List”中列出了当前电路所使用的全部元器件，如电路中还需要添加表中相同的元器件，可直接从“In Use List”中选取。

（2）对电路窗口上的元器件操作。元器件放置到电路窗口后，根据需要还可以对其进行移动、删除、旋转、改变颜色等操作，这些操作可用 Edit（编辑）菜单下的下拉命令来完成，也可以通过右击后出现的快捷菜单中的选项来完成。

（3）元器件参数设置。对于原理图中时钟电压源的参数设置操作如下：选中“时钟电压源”，双击鼠标，系统会弹出“CLOCK_VOLTAGE”对话框，如图 2-8 所示，可以对时钟电压源的相关参数进行设置。

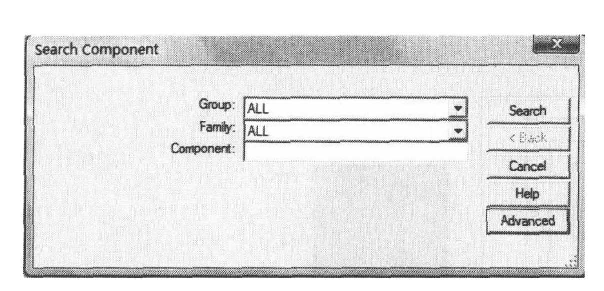


图 2-7　“Search Component”对话框

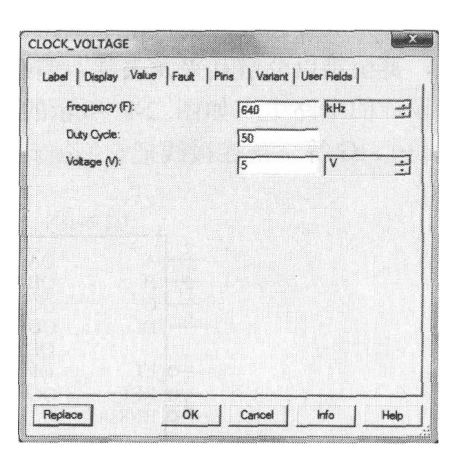


图 2-8　“CLOCK_VOLTAGE”对话框

4．连接线路

对电路窗口放置好的元器件进行线路连接是编辑原理图的重要步骤，Multisim10 软件具有非常方便的线路连接功能，接下来介绍线路的连接方法。

（1）两元器件引脚之间连接。只要将光标移动到引脚附近，就会自动形成一个带十字的圆黑点，这时单击不放并拖动光标，就会自动形成一条虚线，移动光标到要连接的元器件引脚处再单击，一条连接线就完成了。

（2）放置节点。节点即导线与导线的连接点，在图中表示为一个小圆点。一个节点最多可以连接 4 个方向的导线，即上下左右每个方向只能连接一条导线，且节点可以直接放置在连线中。放置节点的方法是：单击“Place\Junction”菜单命令，会出现一个节点跟随光标移动，即可将节点放置在导线上合适的位置。

注意：为了可靠连接，在放置连接点之后，稍微移动一下与该连接点相连的其中一个元器件，查看是否存在脱离现象（即虚焊）。

（3）元器件引脚与线路连接。元器件引脚与线路连接有两种方法：一是从元器件引脚向线路连接；二是由线路上的节点向元器件引脚连接。

① 从元器件引脚向线路连接。先让光标移向元器件的引脚，当到达元器件引脚附近时，就会自动形成一个带十字的圆黑点，这时单击不放并拖动光标，就会自动拖出一条虚线。接着移动光标到要连接的线路之处再单击，会自动的在线路上形成一个节点，元器件引脚与线路的连接就完成了。

② 从线路向元器件引脚连接。从线路向元器件引脚连接，应首先在线路上设置一个节点，然后从这个节点再向元器件引脚连接。

（4）线路之间的连接。有时需要将两条线路之间连接起来，连接的方法是：首先在其中一条线路上合适的位置放置一个节点，然后使光标移动到该节点，既会自动形成一个带十字的圆黑点，这时单击确认，并拖动光标形成一条虚线，当到达要连接线路的合适位置后再单击，就会形成一个新的节点，且这两个节点之间产生了一条连线。

注意：在线路中经常遇到两条交叉而不连接的情况，在进行交叉连接时可直接穿过，不要在线路上不断单击，这样就不会产生连接点。

（5）放置总线。现代电子电路中，集成电路已成为电路的主要器件，电路的连接越来越多，甚至难以分辨其来龙去脉，但经常会遇到多条导线按同一方向连接的情况。例如，我们要设计的电路中有如图 2-9 所示的一个七段显示译码器与七段显示器连接电路，其由 A、B、…、G 等 7 条连接线把它们彼此连接起来。

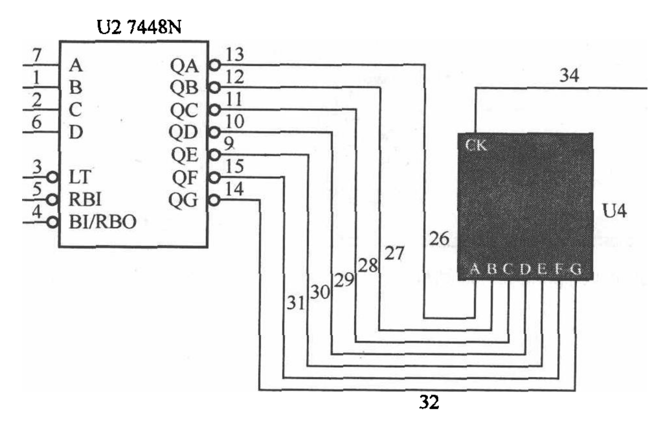


图 2-9　七段显示译码器与七段显示器连接电路

如果利用总线来连接，将两端的单线分别接入总线，构成单线-总线-单线的连接方式，线路就会简单得多。

总线的操作模式有两种，即“Net”模式和“Busline”模式。

在“Net”模式下，一条总线就是一个网络标识的集合，每次当一条总线通过总线入口连接到总线上时，用户可以选择添加这个网络到总线中或者连接到总线中已存的网络上。“Net”模式是一般情况下，绘制总线常用的模式。

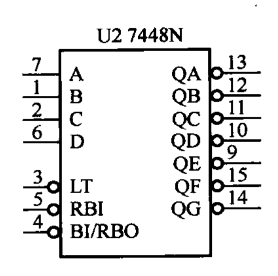
在“Busline”模式下，用户可以预先定义总线中包含的网络的数量和名称。当连接一个导线到总线入口时，用户需要从已存的网络中选择一个进行连接。在多页面电路设计中，如果需要将两个页面的单元电路用总线进行连接，可以用离页总线连接器进行连接。

总线模式的选择可以通过单击“Options\Sheet Properties”命令，弹出“Sheet Properties”对话框，打开“Wiring”选项卡进行设置。

放置总线的操作过程如下。

启动 Place 菜单的 Bus 命令或者单击快捷菜单，进入绘制总线的状态。拖动所要绘制总线的起点，即可拉出一条总线。如要转弯，则单击鼠标即可。达到目的地后，双击即可完成该总线，系统自动给出总线名称“Bus1”，如图 2-10 所示。如要修改总线名称，则双击该总线，打开“Bus1”对话框。在其中 Reference ID 栏内输入新的总线名称，然后单击“确定”按钮。

接着绘制第 1 个元器件各引脚与该总线连接的单线，将光标移向所要连接的元器件引脚，待出现一个带十字的圆黑点时单击，然后拖动鼠标移向总线，当与总线出现斜相交的线时再单击，则出现如图 2-11 所示的对话框。单击“OK”按钮关闭对话框，完成一条导线到总线的连接。用同样的方法按图 2-12 所示步骤完成绘制。



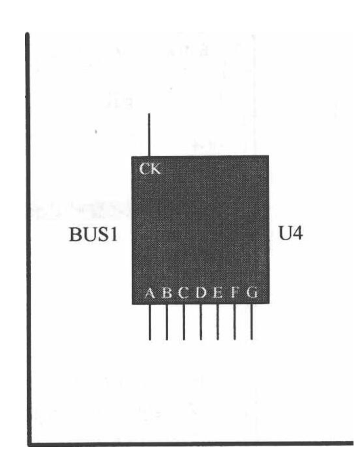


图 2-10 绘制总线操作

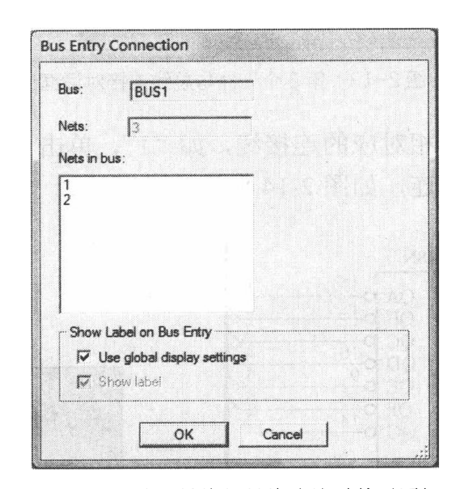


图 2-11 绘制单线与总线连接时的对话框

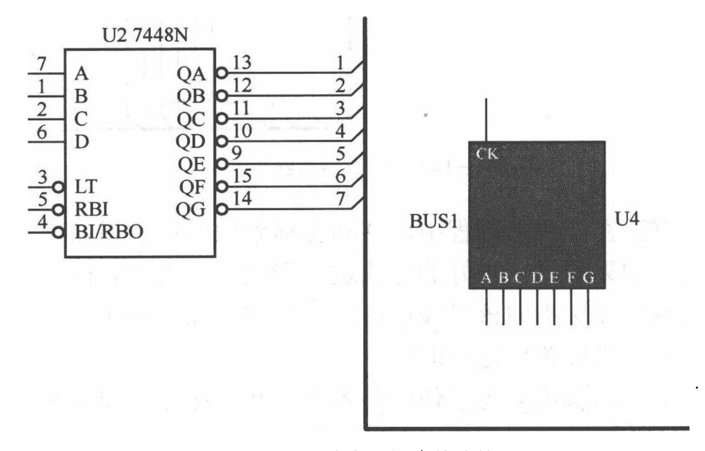


图 2-12 单线与总线的连接

绘制第 2 个元器件与总线连接的单线，将光标移向第 2 个元器件的引脚，待出现一个带十字的圆黑点时单击，然后拖动鼠标移向总线，当与该总线斜交时再单击则出现如图 2-13 所示的对话框。

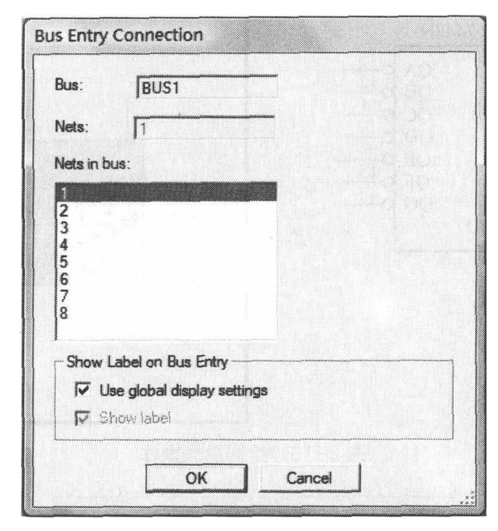


图 2-13　第 2 个元件与总线连接对话框

在对话框的下面栏中选择相对应的连接线，如“1”，单击“OK”按钮。用类似的操作将第 2 个元件的各引脚与总线相连，如图 2-14 所示。

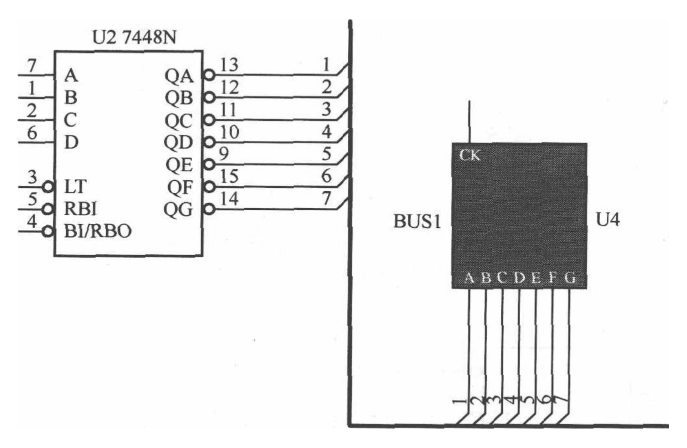


图 2-14　用总线将两器件相连

（6）调整导线位置。在连线的过程中，移动光标到达导线的拐点位置单击，可得到一条自行设定的导线轨迹。如果对已连接好的导线进行调整，可先将光标对准欲调整的导线，单击鼠标将其选中，接着让光标指向该导线单击，就会在导线上形成一个双箭头的调整符，按住鼠标左键移动导线至适当位置后松开即可。

（7）设置连线与节点的颜色。为了使电路各连线及连接点之间彼此清晰可辨，可通过设置不同的颜色来区分，方法是：将光标指向某一连线或节点，右击并在弹出的快捷菜单中单击“Change Color…”命令将打开颜色对话框，选取所需的颜色后单击“确定”按钮，这时连线和节点的颜色将同时改变。

（8）删除连线和节点。如果要删除连线或节点，则将光标指向所要删除的连线或节点，右击，同时也自动弹出快捷菜单，单击“Delete”按钮即可。

2.5.2 虚拟仪器的使用

Multisim10 的虚拟仪器仪表工具栏中共有虚拟仪器仪表 18 台、电流检测探针 1 个、4 种 LabVIEW 采样仪器和动态测量探针 1 个。大多具有和真实仪器仪表相同的面板，用户可根据需要选择，将其调到电路窗口，并与电路连接。在仿真运行时，完成对电路的电压、电流、电阻数值、波形等物理量的测量，用起来几乎和真实仪表的一样。由于仿真仪器的功能是软件化的，所以具有测量数值精确、价格低廉、使用灵活方便的优点。利用这些虚拟的仪器仪表可以对设计的电路进行测试和分析，大大提高了电路的设计效率和设计的准确性。

Multisim10 的虚拟仪器仪表工具栏中除包揽一般电子实验中所常用的测量仪器之外，还拥有一批一般电子实验室中不太可能配置的仪器，其中包括 3 台安捷伦的高档测量仪器，它们分别是：$6\frac{1}{2}$位高性能数字万用表 Agilent34401A；宽带达 100MHz、具有两个模拟通道和 16 个逻辑通道的高性能数字示波器 Agilent54622D；一台宽频带、多用途、高性能的函数信号发生器 Agilent33120A；另外，还有一台美国“泰克”公司的高性能 4 通道数字存储示波器 TDS2040。这四台高性能虚拟测量仪器，不仅功能齐全，而且它们的面板结构、旋钮操作完全和真实仪器一模一样。

除了以上介绍的虚拟仪器之外，还有几台仅用于数字电路实验但一般实验室也不太可能配备的仪器，它们分别是：4 通道示波器、字信号发生器、逻辑分析仪和逻辑转换仪。另外，Multisim10 还允许用户使用 LabView 设计的仪器仪表，在信号测量中，用户还可以根据实际需要设计自己的虚拟仪器，使电路的测量更加简单，并且可以灵活地对仪器仪表进行升级。

1. 仪器仪表工具栏

Multisim10 界面的右侧列出了仪器仪表工具栏，其中包含了用户在电路仿真中将会用到的各种仪器仪表，如图 2-15 所示。

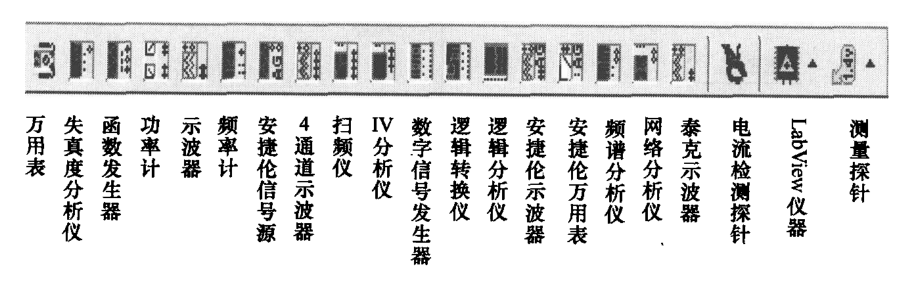


图 2-15 仪器仪表工具栏

Multisim10 虚拟仪器分为如下 6 大类。

（1）模拟（AC 和 DC）仪器。

（2）数字（逻辑）仪器。

（3）射频（高频）仪器。

（4）电子测量技术中的真实仪器，如 Agilent、Tektronix 仪器模拟。

（5）测试探针。

（6）LabVIEW 仪器。

2．操作虚拟仪器

（1）单击仪器栏中需要使用的仪器图标后松开，移动鼠标到电路设计窗口中，再单击，仪器图标变成了仪器接线符号在设计窗口中显示。

（2）双击仪器接线符号即可弹出仪器面板。可以在测试前或测试中更改仪器面板的相关设置，如同实际仪器一样。更改仪器面板的设置，一般是更改量程、坐标、显示特性、测量功能等。仪器面板设置的正确与否对电路参数的测试是非常关键的，如果设置不正确很可能会导致仿真结果出错或是难以读数。

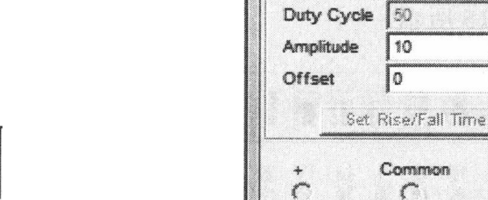
（3）“激活”电路。单击仿真按钮进行仿真。电路进入仿真，与仪器相连的电路上那个点上的电路特性和参数就被显示出来了。

当设计电路被仿真时，可以改变电路中元器件的标值，也可以调整仪器参数设置，但在有些情况下必须重新启动仿真，否则显示的一直是改变前的仿真结果。下面以“函数信号发生器”为例，进行操作设置。

3．函数信号发生器的使用

（1）函数信号发生器的接线图标及面板。函数信号发生器是一个产生正弦波、三角波和方波信号的电压源，函数信号发生器接线图标如图 2-16（a）所示。

双击函数信号发生器接线图标即出现其面板图，如图 2-16（b）所示。



（a）函数信号发生器接线图标　　（b）函数信号发生器面板

图 2-16　函数信号发生器图标和面板

函数信号发生器面板上部的 3 个信号波形选择按钮，用于选择仪器产生波形的类型。中间的几个选项窗口，分别用于选择产生信号的频率、占空比、信号幅度和直流偏置。

函数信号发生器面板下部的 3 个接线端子，COM 端连接电路的参考地点，“+”为正波形端，“–”为负波形端。

（2）连接。函数信号发生器的图标有“+”、“Common”和“–”，3 个输出端子与外电路相连输出电压信号，其连接规则如下。

① 连接“+”和“Common”端子，输出信号为正极性信号，峰-峰值等于两倍幅值。

② 连接“Common”和“–”端子，输出信号为负极性信号，峰-峰值等于两倍幅值。

③ 连接“+”和“-”端子，输出信号的峰-峰值等于 4 倍幅值。

④ 同时连接“+”、“Common”和“-”端子，且把 Common 端子与公共地（Ground）符号相连，则输出两个幅值相等、极性相反的信号，即差分信号。

（3）面板操作。改动面板上的相关设置，可改变输出电压信号的波形类型、大小、占空比或偏置电压等。

Waveforms 区：选择输出信号的波形类型，有正弦波、三角波和方波 3 种周期性信号供选择。

Signal Options 区：对 Waveforms 区中选择的信号进行相关参数设置。

① Frequency：设置所要产生信号的频率。

② Duty Cycle：设置所要产生信号的占空比，该参数设置只对三角波和方波有效，对于正弦波信号不起作用。可调范围为 1%～99%。

③ Amplitude：设置所要产生信号峰值，与信号直流偏置有关。

④ Offset：设置偏置电压值，即把正弦波、三角波或方波叠加在设置的偏置电压上输出。

Set Rise/Fall Time 按钮：设置所要产生信号的上升时间与下降时间，而该按钮只有在产生方波时有效。单击该按钮后，出现如图 2-17 所示的对话框。

此时，请在栏中以指数格式设定上升时间（或下降时间），再单击“Accept”按钮即可。如单击“Default”按钮，则恢复默认值 1.000000e–12。

（4）案例：观察函数发生器的波形。

在电路图中，放置一个函数发生器和一个示波器，连接方法如图 2-18 所示。

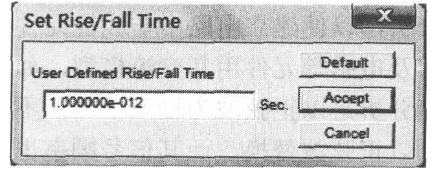


图 2-17　上升时间与下降时间设置对话框

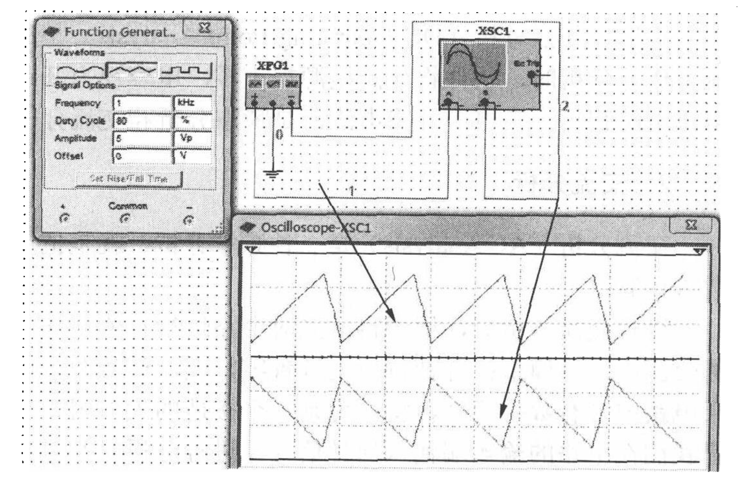


图 2-18　观察函数发生器产生信号

操作方法：

① 设置函数发生器的功能，察看产生的信号波形。

② 更改信号参数，察看产生的信号波形。

③ 设置为三角波或方波时，调整占空比，察看产生信号的波形，同时加深理解信号的正向和反向的概念。

4. 使用虚拟仪器的注意事项

使用虚拟仪器的注意事项如下。

（1）在仿真过程中电路的元器件参数可以随时改变，也可以改变接线。

（2）一个电路中可允许多个不同仪器或多个同样的仪器同时使用。

（3）可以以电路文件方式对某一测试电路的仪器设置与仿真数值进行保存。

（4）可以在图标仪上改变仿真结果显示。

（5）可以改变仪器面板的大小。

（6）仿真结果很容易以“.TXT”、“.LVM”和“.TDM”形式输出。

2.5.3 基本分析方法

Multisim10 软件具有 18 种仿真分析方法，这些方法对于电路分析和设计都非常有用，学会这些方法，可以增加分析电路、设计电路的能力。仿真分析主要解决两个方面的问题：一是认识各种仿真分析的功能，也就是分析什么问题；二是设置正确的分析参数，也就是怎样完成分析。一般情况下，计算的数据区间又称为扫描区间，计算数据点的间隔称为步距。而计算机需要的计算变量为节点电压、流过某个元件的电流以及功率。

本小节只对这些分析方法做简单的介绍，对于想进一步了解的读者，可以参考 Multisim 软件的帮助文档。

1. 直流工作点分析

直流工作点分析（DC Operating Point Analysis）是进行电路其他分析的基础，在进行直流工作点分析时，软件将交流电源视为零，电容视为开路，电感视为短路。求得电路的每个节点电压和每一条电源支路电流，是瞬态分析和交流分析的基础。

2. 交流分析

交流分析（AC Analysis）是一种频域分析，就是把用户指定的交流输出量作为频域的函数来计算。

进行交流分析时，程序自动先对电路进行直流工作点分析，以便建立电路中非线性元件的交流小信号模型；并把直流电源置零，交流信号源、电容及电感等元件用其交流模型，如果电路中含有数字元件，将认为是一个接地的大电阻。交流分析是以正弦波为输入信号，不管我们在电路的输入端输入何种信号，进行分析时都将自动以正弦波替换，而其信号频率也将以设定的范围替换之。交流分析的结果，以幅频特性和相频特性两个图形显示。如果将扫描仪连至电路的输入端和被测节点，也可获得同样的交流频率特性。

3. 瞬态分析

瞬态分析（Transient Analysis）指对所选定的电路节点进行的时域响应分析，即观察节点在整个显示周期中每一时刻的电压波形。在进行瞬态分析时，直流电源具有恒定的数值，交流电源的数值随时间而变化，电容和电感都具有储能的特性。分析时，电路的初始状态可由用户自行指定，也可由程序自动进行直流分析，用直流解作为电路初始状态。瞬态分析的结

果通常是分析节点的电压波形，和用示波器可观察到相同的结果。

4. 直流扫描分析

直流扫描分析（DC Sweep Analysis）是计算电路中某一节点上的直流工作点随电路中一个或两个直流电源的数值变化的情况。利用直流扫描分析，可快速地根据直流电源的变动范围确定直流工作点。它的作用相当于每变动一次直流电源的数值，则对电路作几次不同的仿真。

注意：如果电路中有数字器件，可将其当作一个大的接地电阻处理。

5. 参数扫描分析

参数扫描分析（Parameter Sweep Analysis）可以较快地考查电路元器件数值在一定范围变化时对电路的影响，这种分析方法相当于给该元器件每次取不同的值，进行多次仿真。观察变化时对电路直流工作点、瞬态特性及交流频率特性的影响，以便对电路的某些性能指标进行优化。

6. 温度扫描分析

温度扫描分析（Temperature Sweep Analysis）是研究温度变化对电路性能的影响，通常电路的仿真都是假设在27℃下进行的，而由于许多电子器件与温度有关，当温度变动时，电路的特性也会产生一些改变。该分析相当于在不同的工作温度下多次仿真电路性能。不过，Multisim 中的温度扫描分析并不是对所有元件都有效，仅限于一些半导体器件和虚拟电阻。

7. 傅里叶分析

傅里叶分析（Fourier Analysis）是分析非正弦周期信号的一种方法，使用该方法可以清楚地知道周期性非正弦信号中的直流分量、基波分量和各次谐波分量的大小。傅里叶分析可以给出幅度频谱和相位频谱。

8. 噪声分析

噪声分析（Noise Analysis）是分析噪声对电路的影响。该分析提供热噪声、散粒噪声和闪烁噪声模型，在分析中假设各个噪声源互不相关，总噪声为每个噪声源对指定输出节点产生的噪声均方根的和。输出图形为输入噪声功率和输出噪声功率随频率变化的曲线。

9. 噪声系数分析

噪声系数分析（Noise Figure Analysis）是指分析输入信噪比/输出信噪比。信噪比是一个衡量电子线路信号质量好坏的参数，噪声系数分析用来衡量有多大的噪声加入到信号中。

10. 失真分析

失真分析（Distortion Analysis）用于分析小信号电路的谐波失真，研究瞬态分析中不易观察到的小失真。

11．灵敏度分析

灵敏度分析（Sensitivity Analysis）用于计算电路某一节点对电路中元器件参数的灵敏度，以便找到灵敏度大的元件。该分析可以进行直流灵敏度和交流灵敏度计算，对于直流灵敏度直接给出灵敏度数值，对于交流灵敏度则给出相应的曲线。

12．极点—零点分析

极点—零点分析（Pole-Zero Analysis）用于求解交流小信号电路传递函数中的极点和零点的数值，以此提供电路稳定性的判断依据。分析结果为极点和零点的数值。

13．传递函数分析

传递函数分析（Transfer Function Analysis）是在直流小信号状态下，求解电路的输入阻抗、输出阻抗和传递函数。在分析中，要求在输出端指定输出节点或流过某一元器件上的电流，还要求在输入端指定一个独立电源。该分析直接给出分析数值。

14．最坏情况分析

最坏情况分析（Worst Case Analysis）是统计方法中的一种，用于分析电路中元件参数在其容差边界点上取某种组合时的电路性能。

15．蒙特卡罗分析

蒙特卡罗分析（Monte Carlo Analysis）用于分析电路性能与元件容差之间的关系，该方法基于给定电路元件参数容差规律，随机抽取一组带有容差的元件参数后，对电路进行直流、交流或瞬态分析，以统计的方法估算电路的性能。

16．线宽分析

线宽分析（Trace Width Analysis），当仿真电路在 Multism10 中完成仿真分析，并达到各项参数要求时，就可以进行设计 PCB 的工作。线宽分析就是用来确定在设计 PCB 时所能允许的最小的导线宽度。

17．批处理分析

批处理分析（Batched Analysis）就是将同一个仿真电路的不同分析组合在一起执行的分析方式。

18．用户自定义分析

用户自定义分析（User Defined Analysis）就是由用户通过 SPICE 命令来定义某些仿真分析的功能，以达到扩充仿真分析的目的。

19．分析方法应用举例

例：应用 Multisim10 对图 2-19 所示的单级晶体管放大电路进行直流、交流、参数扫描(R_2)

和瞬态分析。

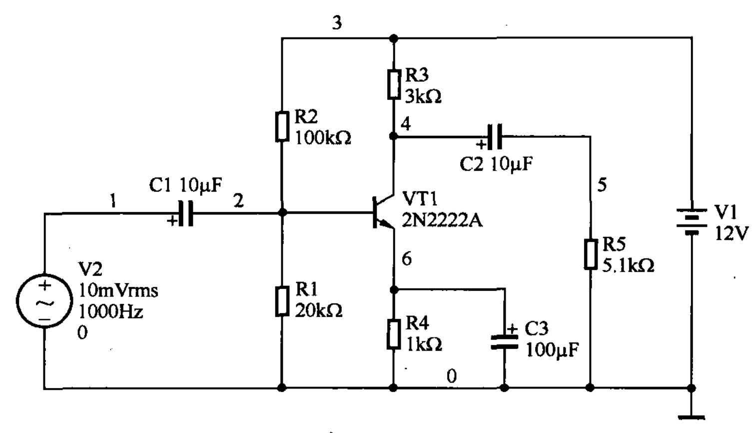


图 2-19　晶体管放大电路

（1）仿真分析内容。

① 直流工作点分析。计算各节点电压和支路电流。

② 交流分析。采用每 10 倍频程扫描 10 点的方式分析 1Hz～10GHz 内电路的频率特性。

③ 瞬态分析。同时用示波器观察输入输出波形并进行比较。

④ 参数扫描分析。R_2 以 20kΩ为增量从 80kΩ线性增长到 120kΩ，分析电路的瞬态特性。

（2）仿真分析。仿真结果的观察可分为两类：一类是从仿真仪表上观察仿真结果，开始仿真时可直接单击仿真电源开关使之闭合，就像在实验室使用仪表测试电路，观察结果一样；另一类是从分析设置的运行来看结果，仿真开始按下仿真设置窗口中的“Simulate”按钮，仿真运行完成后，系统直接进入 Analysis Graphs 观察仿真结果，此时仿真结果一般是以曲线显示的。例如，运行 AC 分析，便可得到频率特性曲线，如同用逐点法测试数据，画在坐标纸上。

① 直流工作点分析。单击“Simulate/Analysis/DC Operating Point Analysis”菜单命令。在“Output”选项下，从“Variable in Circuit”栏选中所有变量，单击“Add”按钮加到“Selected variables for analysis”栏中，单击“Simulate”按钮进行直流工作点分析，得到如图 2-20 所示的结果。

	DC Operating Point	
1	V(4)	8.27983
2	V(2)	1.87323
3	V(3)	12.00000
4	V(6)	1.24766
5	V(1)	0.00000
6	V(5)	0.00000
7	I(v1)	-1.34132 m
8	I(v2)	0.00000

图 2-20　直流工作点分析结果

② 交流分析。单击“Simulate/Analysis/AC Analysis”菜单命令，在出现的对话框中进行参数设置。

在“Frequency Parameters”下设置参数如下：

```
Start frequency: 1Hz;
Stop frequency: 100GHz;
Sweep Type: Decade;
Number of points per decade: 10;
Vertical Scale: Linear。
```

在“Output”下设置参数为：从“Variables in circuit”栏选择需要分析的输出节点 5，这样输出节点的仿真结果就可显示在“Analysis View”窗口中。

参数设置好后单击“Simulate”按钮，进行交流分析，得到如图 2-21 所示的输出结果，包括幅频特性和相频特性。

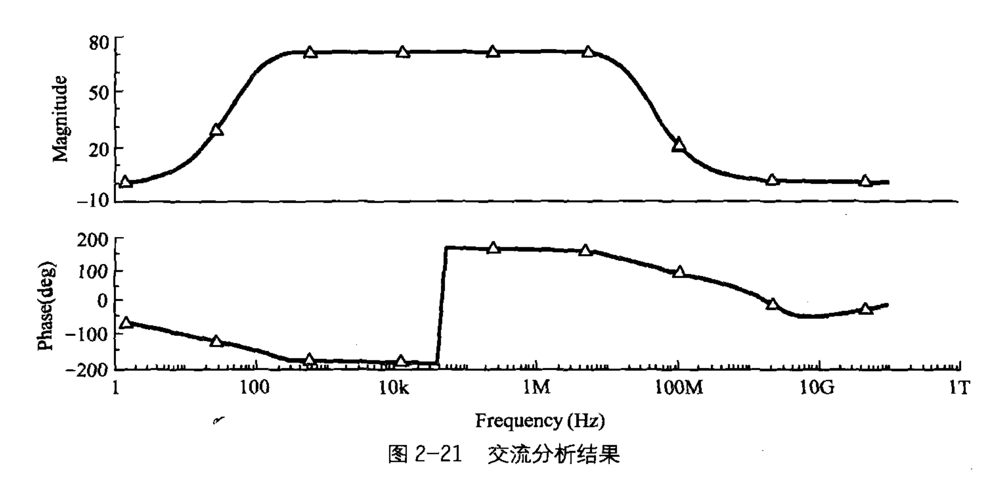


图 2-21 交流分析结果

③ 瞬态分析。单击“Simulate/Analysis/Transient Analysis”菜单命令，在“Analysis Parameters”下设置参数如下：

```
Initial conditions: Automatically determine…;
Start time: 0;
End time: 0.002s;
Maximum time step settings: Generate time steps automatically。
```

同样，选择好分析节点后单击“Simulate”按钮，进行瞬态分析，得到如图 2-22 所示的输出结果。

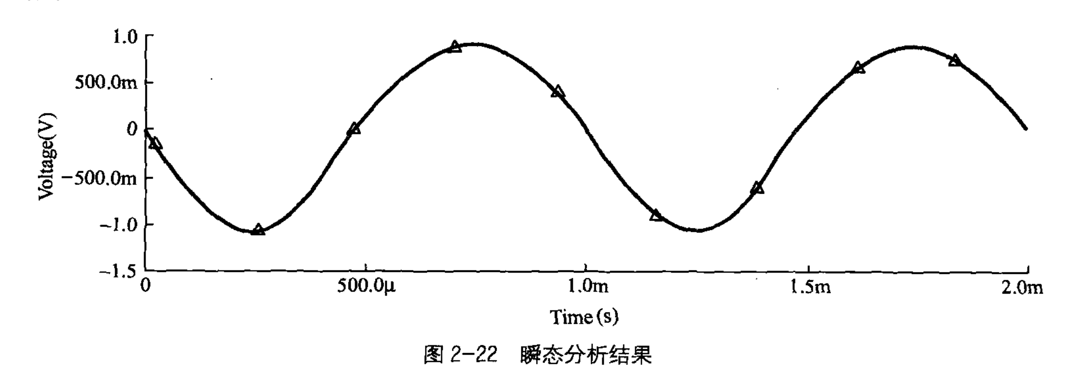


图 2-22 瞬态分析结果

④ 参数扫描分析。单击“Simulate/Analysis/Parameter Sweep”菜单命令，在“Analysis Parameters”下设置参数如下：

```
Sweep Parameter: Device parameter;
Device Type: Resistor;
Name: rr2;
Parameter: Resistance;
Sweep Variation Type: Linear;
Start: 80 000;
Stop: 120 000;
# of point: 3;
Analysis to sweep: Transient analysis。
```

然后，选中“Group all traces on one”选项，单击“Edit Analysis”按钮，进入“Transient Analysis”设置对话框，参数设置同“③”，然后单击“OK”按钮，返回“Parameter Sweep”设置对话框，单击“Simulate”按钮，进行参数扫描分析，得到如图 2-23 所示的输出结果。

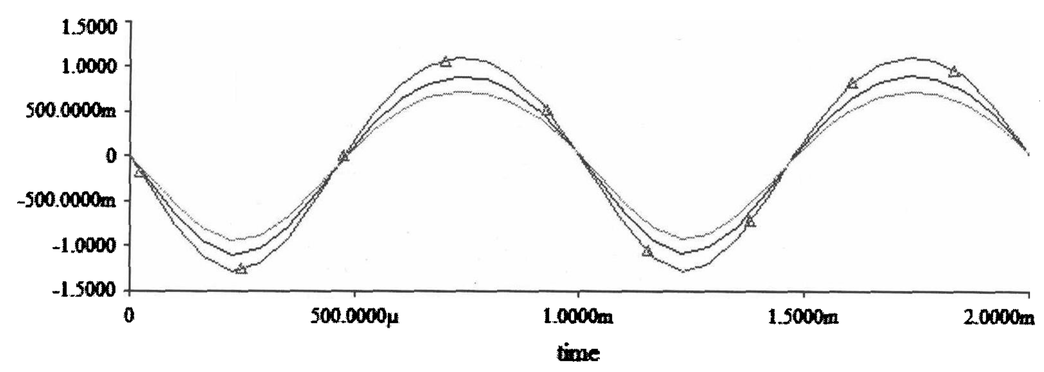


图 2-23　参数扫描分析结果

2.6　综合设计与仿真

使用 Multisim 软件可以仿真电路，帮助分析和学习电路，更重要的是帮助电路设计者验证设计结果，搭配元件参数。在我们做实际实验之前，使用 Multisim 软件对要实验的电路进行仿真验证，即可加深对电路原理的理解，又可节省资金和时间。另外，若实际实验的条件不具备，对某些课程设计的结果也可以只用 Multisim 软件仿真验证。下面介绍 Multisim 软件仿真实例。

2.6.1　反相比例运算电路分析

利用理想的运算放大器组成的反相比例运算电路，如图 2-24 所示。

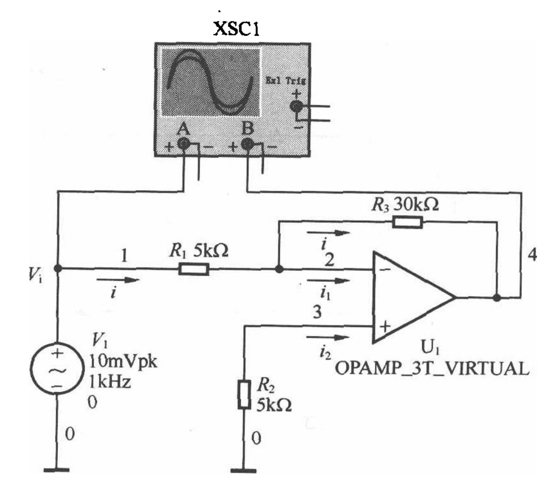


图 2-24　反相比例运算电路

1．分析

根据虚断，$i_1\approx0$，$i_1\approx i_2$，故 $V_+\approx0$。

根据虚短，$V_+\approx V_-\approx0$。

$i=(V_i-V_-)/R_1\approx V_i/R_1$。

$V_o\approx-i\times R_f=-V_i\times R_f/R_1$。

所以，可得电压增益 $A_{vf}=V_o/V_i=-R_3/R_1=-6$。

根据上述关系式，该电路可用于反相比例运算。下面通过 Multisim 对该电路进行仿真。

2．操作方法

（1）按图 2-24 所示，连接电路。

（2）单击 Multisim 仿真按钮，通过示波器观察输入和输出波形，结果如图 2-25 所示。

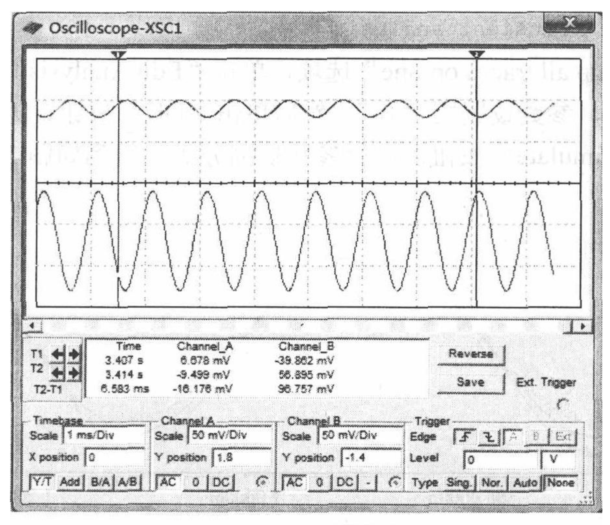


图 2-25　仿真结果

（3）从图 2-25 所示的仿真结果中可以看出，输出和输入波形的相位相反，当游标在如图所示的位置时，$A_{vf}=-39.862/6.678=-5.969$，这与理论分析结果是一致的。

（4）单击“Simulate/Analysis/AC Analysis”菜单命令，弹出“AC Analysis”对话框。

（5）设置交流分析的扫描频率为“1Hz～1GHz”；扫描类型为“Decade”；“Number of points per decade”为“10”；“Vertical scale”为“Logarithmic”。

（6）设置分析变量为“V(4)”。

（7）执行交流分析，通过分析结果查看电路的幅频特性和相频特性，如图 2-26 所示。

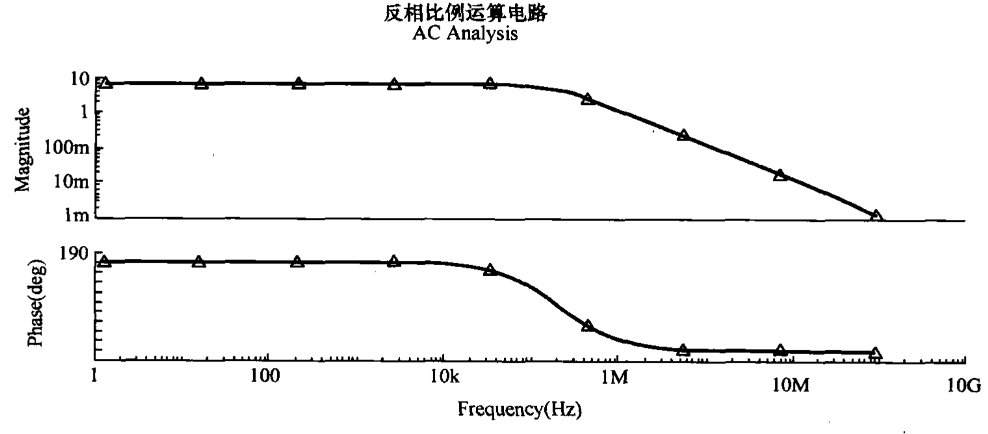


图 2-26　交流分析结果

同理，还可以分析同相输入和差分输入。

2.6.2 三路智力竞赛抢答器仿真设计

1. 设计要求

（1）制作一个可容纳三组参赛者的竞赛抢答器，每组设置一个抢答器按钮，分别为“[1]”、“[2]”、“[3]”，供参赛者使用。

（2）电路应具有第一抢答信号的鉴别和锁存功能。在主持人用“[X]”键清零、发出抢答指令后，如果某参赛者在第一时间按动抢答开关成功，应立即将其输入锁存器自锁，使其他组别的抢答信号无效，并用编码、译码及数码显示出该组参赛者的组号。

（3）若同时有两组或两组以上信号抢答，则所有的抢答信号无效，显示器显示“0”。

2. 三路智力抢答器仿真设计

（1）原理框图。由设计要求“若同时有两组或两组以上信号抢答，则所有的抢答信号无效，显示器显示 0 字符”，所以抢答信号处理须使用触发器。为此，用 1 块四 D 触发器 74LS11N（上升沿触发）、2 个 3 输入与非门 74LS10N、1 个 3 输入与门 74LS11N、1 块 555 定时器、1 块共阴七段数码显示器和若干按钮开关、电阻器、电容器等器件设计制作一个可容纳[1]、[2]、[3]三个组别参加的三路智力抢答器（+5V 电源另配），设计原理框图如图 2-27 所示。

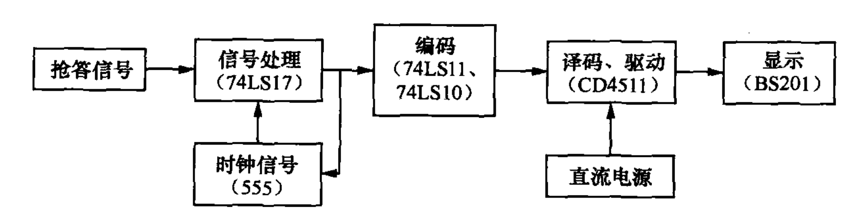


图 2-27 三路智智力抢答器设计原理框图

（2）逻辑赋值、原理图。将如图 2-27 所示的原理框图设计细化，有三路智力抢答器原理图，如图 2-28 所示。图中[X]为主持人清零信号，低电平有效，清零后，显示器清零；然后，单击开关控制键[X]，接入高电平，抢答开始；[1]、[2]、[3]分别为参赛组的组别符号，每一组有一个对应的抢答按钮，抢答按钮高电平有效；抢答时，第一时间抢答成功者的组别符号被显示器显示。

（3）设计说明。当主持人按下清零按钮后，四 D 触发器 U1 的 $\overline{CLR}=0$，被清零，$Q_1=Q_2=Q_3=0$、$\overline{Q_1}=\overline{Q_2}=\overline{Q_3}=1$，显示器 U11 显示 0 字符，所以抢答器无效。

第一抢答信号的鉴别和锁存功能由四 D 触发器 U1，三输入与门 U2A、U3B 和一个用一块 555 定时器电路 U9 构成的多谐振荡器组合完成。

为保证参赛者在按抢答器按钮的瞬间，时钟脉冲信号能适时地到达，由 U9LM555CN 构成的多谐振荡器的振荡周期 $T\approx0.7(R_5+2R_6)C_1=0.7\times(4+2\times3)\times10^3\times10^{-7}\text{s}\approx0.7\text{ms}$，约为 1.4kHz，远高于参赛选手的抢答频率。当主持人命令开始抢答后，U1 的 $\overline{CLR}=1$。此时，抢答有效，设第一组参赛者在第一时间按下了抢答器按钮[1]，U1 的 $\overline{Q_1}=0$，三输入与门 U2A 的输出为 0，U3B 的输出为 0，即 U_1(74LS175)的时钟脉冲信号 CLK（上升沿有效）为 0，被封锁，从而使其后的抢答信号无效。

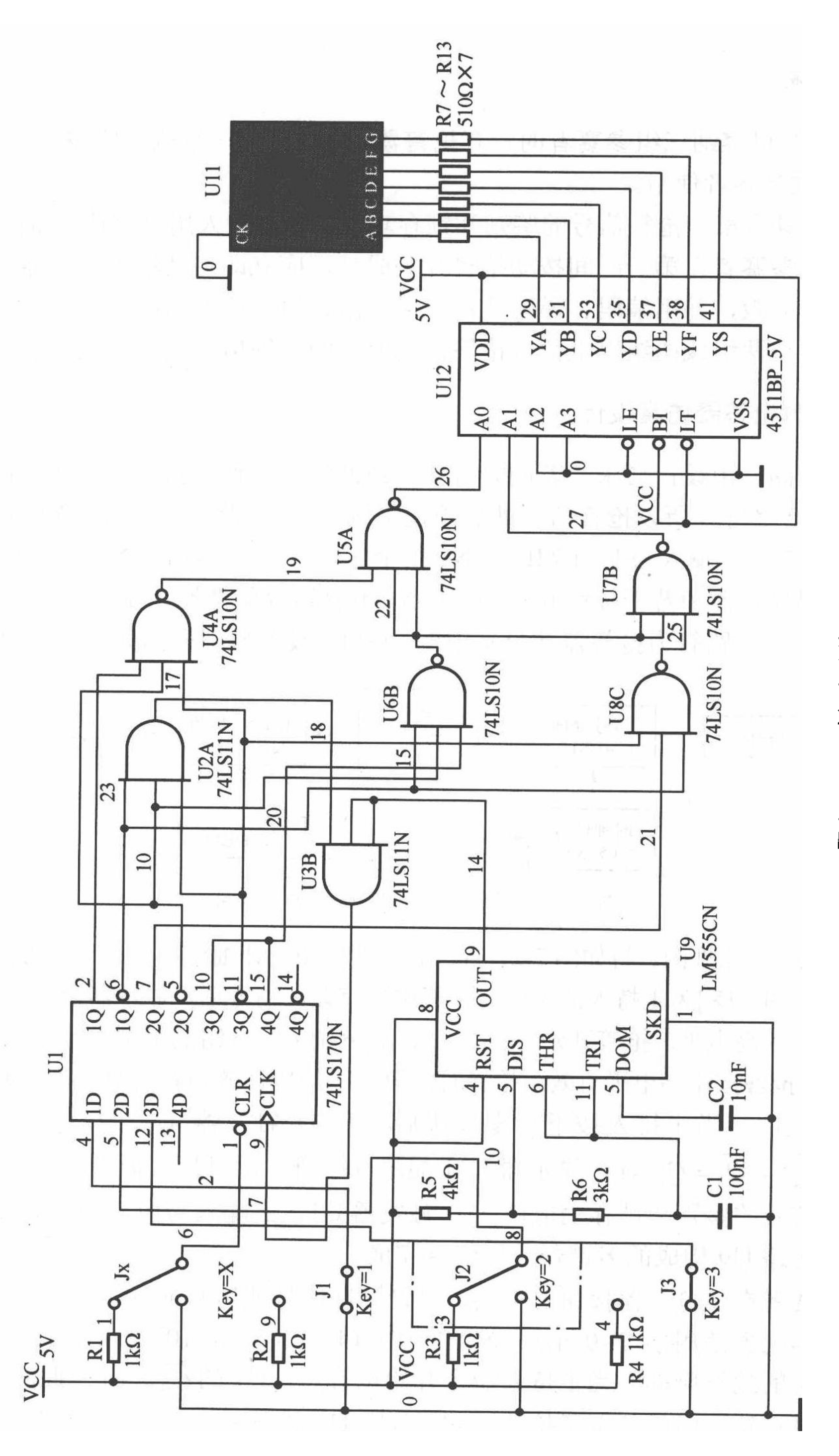


图 2-28 三路智力抢答器

按要求，七段数码显示器译码/驱动电路 U12（CD4511）对应的译码表如表 2-1 所示。

表 2-1　　七段译码显示电路的译码表

四 D 触发器 74LS175 输出			译码/驱动电路 CD4511 输入				显示器对应显示
Q3	Q2	Q1	D	C	B	A	
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	1
0	1	0	0	0	1	0	2
0	1	1	0	0	0	0	0
1	0	0	0	0	1	1	3
1	0	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	0

由表 2-1 可写出七段数码显示器译码/驱动电路 CD4511 输入端 D、C、B、A 对应的逻辑表达式：

$$A=\overline{Q_3}\,\overline{Q_2}Q_1+Q_3\overline{Q_2}\,\overline{Q_1}=\overline{\overline{\overline{Q_3}\,\overline{Q_2}Q_1}\cdot\overline{Q_3\overline{Q_2}\,\overline{Q_1}}}$$

$$B=\overline{Q_3}Q_2\overline{Q_1}+Q_3\overline{Q_2}\,\overline{Q_1}=\overline{\overline{\overline{Q_3}Q_2\overline{Q_1}}\cdot\overline{Q_3\overline{Q_2}\,\overline{Q_1}}}$$

$$C=D=0$$

根据上述逻辑表达式，可用 U4A、U5A、U6B、U7B 和 U8C 构成编码电路，如图 2-28 电路中的对应部分所示。

3．仿真分析

按设计要求进行仿真实验、分析，验证设计功能要求。工作时，主持人需先将抢答器的清零按钮[X]接低电平后，再接高电平（发出“开始抢答”的指令），则第一时间抢答成功的参赛者的组别符号被显示器显示，此后，其后的抢答信号应无效；若同时有两组或两组以上抢答，则所有的抢答信号无效，显示器显示字符 0。

4．电路扩展训练

试设计一个将上述实例中的编码电路由一个 3 线–8 线译码器 74LS138 和一个三输入与非门 74LS10N 构成的三路智力抢答器。

2.6.3　24 小时制多功能数字钟设计

1．技术指标

（1）设计一个具有时、分、秒的十进制数字显示（小时从 00～23）计数器。

（2）具有手动校时、校分功能。

（3）具有整点报时的功能，应该是每个整点完成相应点数的报时，例如 5 点钟响五声。

2．数字时钟功能分析

数字钟是工作生活中常用的电器，其基本的功能是能够准确地显示时、分、秒时间信息。数字钟完整地显示时、分、秒信息需要 6 个数码管，要分别实现时、分、秒的计时需要一个十二进制计数器和两个六十进制计数器；要实现校时功能，需要分别针对时、分、秒的校时电路；要实现 1Hz 的秒钟计数需要时钟振荡电路。根据以上分析数字钟设计由数码显示器，六进制和十二进制计数器、时钟振荡器和校时电路等几部分组成。

分、秒计数需要的六十进制计数器可以由十进制和六进制的计数器串联而成；小时计数需要的十二进制计数器也可以由两个十进制计数器串联加上必要的反馈置零电路设计完成；1Hz 的时钟信号可以由晶振分频后提供，也可以由 555 定时来产生脉冲并分频为 1Hz 后提供。

各单元电路实现后，根据数字钟的基本原理，可以很容易地完成整个数字钟电路的设计。将秒计数器的进位端连接到分计数器的时钟信号输入端，将分计数器的输出进位端连接到小时计数器的时钟信号输入端，显示数码管接到计数器的计数输出端；校时电路连接到时计数器个位、分计数器十位，校分电路连接到分计数器个位、秒计数器十位的时钟端；再通过一个比较器，比较时计数器和响声计数器，在时间出现整点前数秒内，数字钟会自动报时，以示提醒，这样就可以完成整个数字钟的设计。数字钟电路的结构框图如图 2-29 所示。

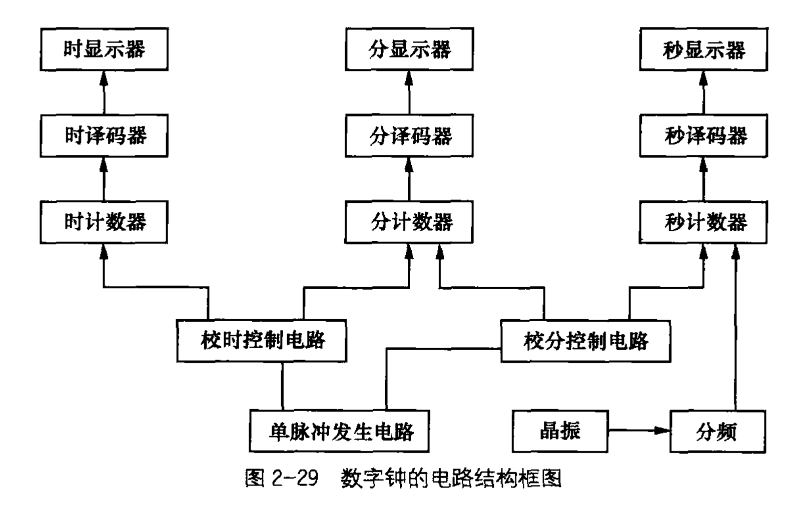


图 2-29　数字钟的电路结构框图

3．数字钟各单元电路设计

（1）时钟振荡电路。取出 LM555 芯片和 3 个电阻、两个电容完成时钟振荡电路电路，需要注意的是，C1 电容为抗干扰电容，设计电路前要根据相关频率计算公式计算 R1、R2、C 的具体取值并通过示波器的显示进行微调。

555 多谐振荡器产生 1kHz 仿真电路和仿真波形图分别如图 2-30 和图 2-31 所示。

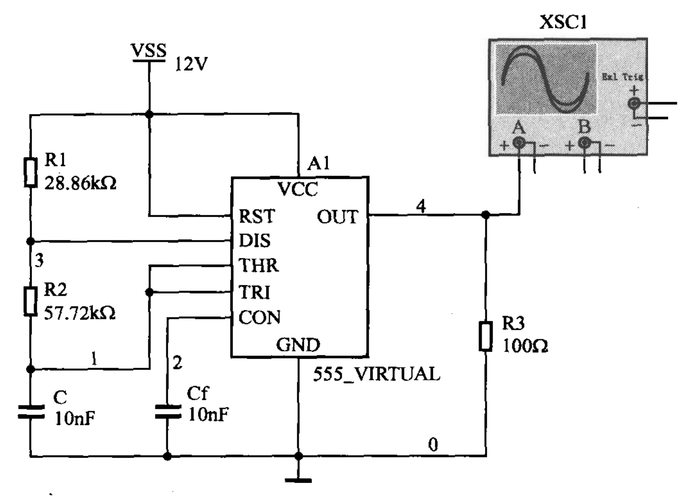


图 2-30　555 多谐振荡器产生 1kHz 仿真电路

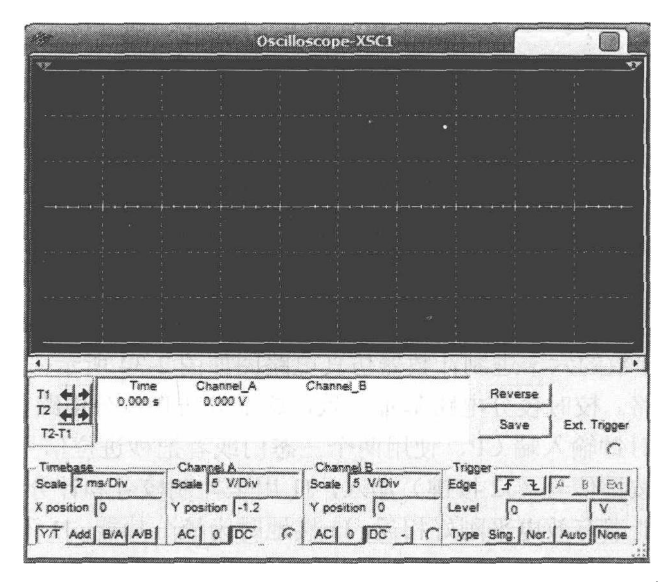


图 2-31　555 多谐振荡器产生 1kHz 仿真波形图

（2）秒脉冲产生电路。74LS160N 在级联使用时一定注意两个时能端 EP、ET 的设置，只有两个均为高电平才计数，当第三级控制时一定是第一级和第二级共同控制第三级，如图 2-32 所示。其中，第二级的 EP 和 ET 一同由第一级的 RCO 控制，第三级的 ER 由第二级的 RCO 控制，而 ET 由第一级的 RCO 控制，这样保障了向第三级的进位是前两个芯片均计满溢出时。如图 2-32 所示是给 555 多谐振荡器产生的 1KHz 分频使用的，使用十进制计数器 74LS160N 计数。

（3）小时计数——二十四进制电路仿真。本例采用本章已经使用过的 74LS160D 完成，首先设计出一百进制的计数器，在 24（00100100）处直接取出所有为 1 的端口，给所有为 0 的端口加非门 74LS00，然后经过与非门后给清零端，使用清零法完成二十四进制，计数范围为 0～23。小时计数的二十四进制计数器仿真电路图如图 2-33 所示。

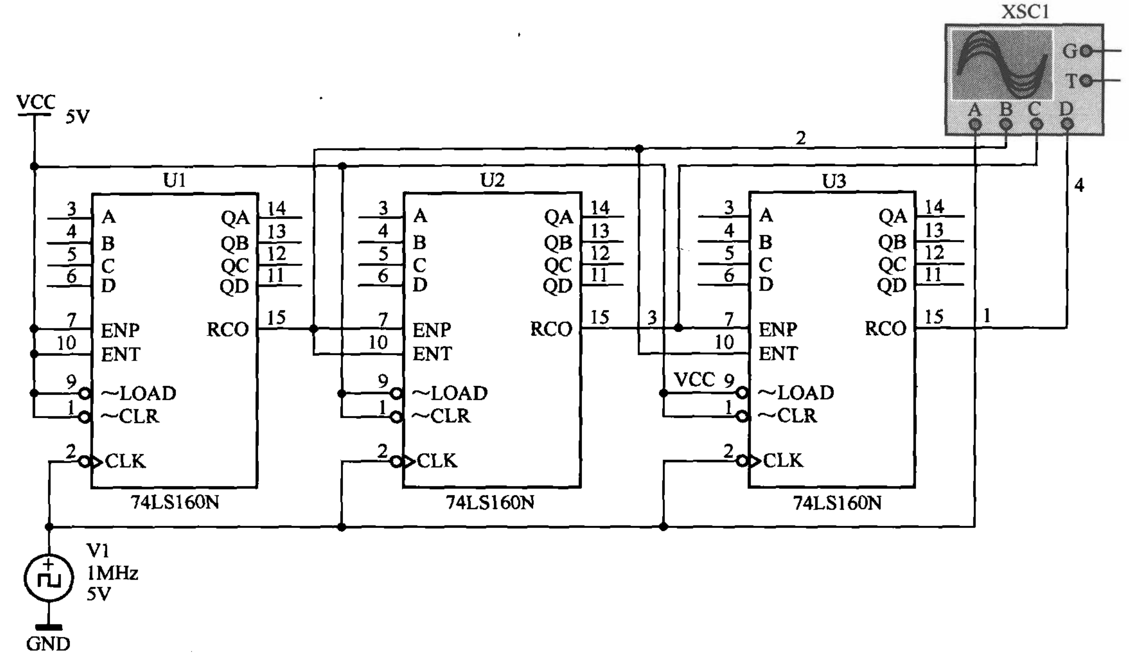


图 2-32　千分频秒脉冲产生仿真电路

（4）分钟/秒计数——六十进制电路仿真。本例类似于二十四进制计数器，采用 74LS160N 完成。首先设计出一百进制的计数器，在 60（01100000）处直接取出所有为 1 的端口，给所有为 0 的端口加非门 74LS00，然后经过与非门后给清零端，使用清零法完成六十进制，计数范围为 0～59。分钟计数的六十进制计数器仿真电路图如图 2-34 所示。

（5）校时校分电路。校时校分电路基本一致，这里只仿真校分电路，如图 2-35 所示，方法是控制六十进制的时钟输入端 CP，使用两个三态门或者把秒进位信号（V1 信号源仿真）加入，或者把校分的按键信号（J2 按键）加入，J1 用来控制校分和计分切换，由于两个三态门 U1A 和 U2A 的使能端有效电平刚好相反，J1 接地时为校分功能，J1 不接地时为计分功能。校时电路与此电路基本一致。

（6）整点报时电路。整点报时电路如图 2-36 所示，包括报时计数电路、停止报时控制电路和蜂鸣器 3 部分。其中，报时计数电路由两个可逆十进制计数器 74LS192 组成，U1、U2 的输出分别连接时计数器的十位和个位输入端，在分进位信号触发下，分计时电路保存当前小时数，并开始递减计数，一直减到 0，停止计数控制电路经过逻辑电路判断给出 0 信号，封锁与门，阻止蜂鸣器工作，停止报时。

4．数字钟整体电路仿真

整体电路可以通过建立主电路，然后为总电路添加子电路的方式完成。

在电路中接入子电路之前，需要给予电路添加输入/输出结点，当子电路被嵌入主电路时，该结点会出现在子电路的符号上，以便使设计者能看到接线点。

当子电路建立好时，新建一个主电路图，单击“Place\New Subcircuit”菜单命令，在出现的对话框中为子电路输入名称。单击“OK”按钮，子电路的影子跟随鼠标移动，在主电路

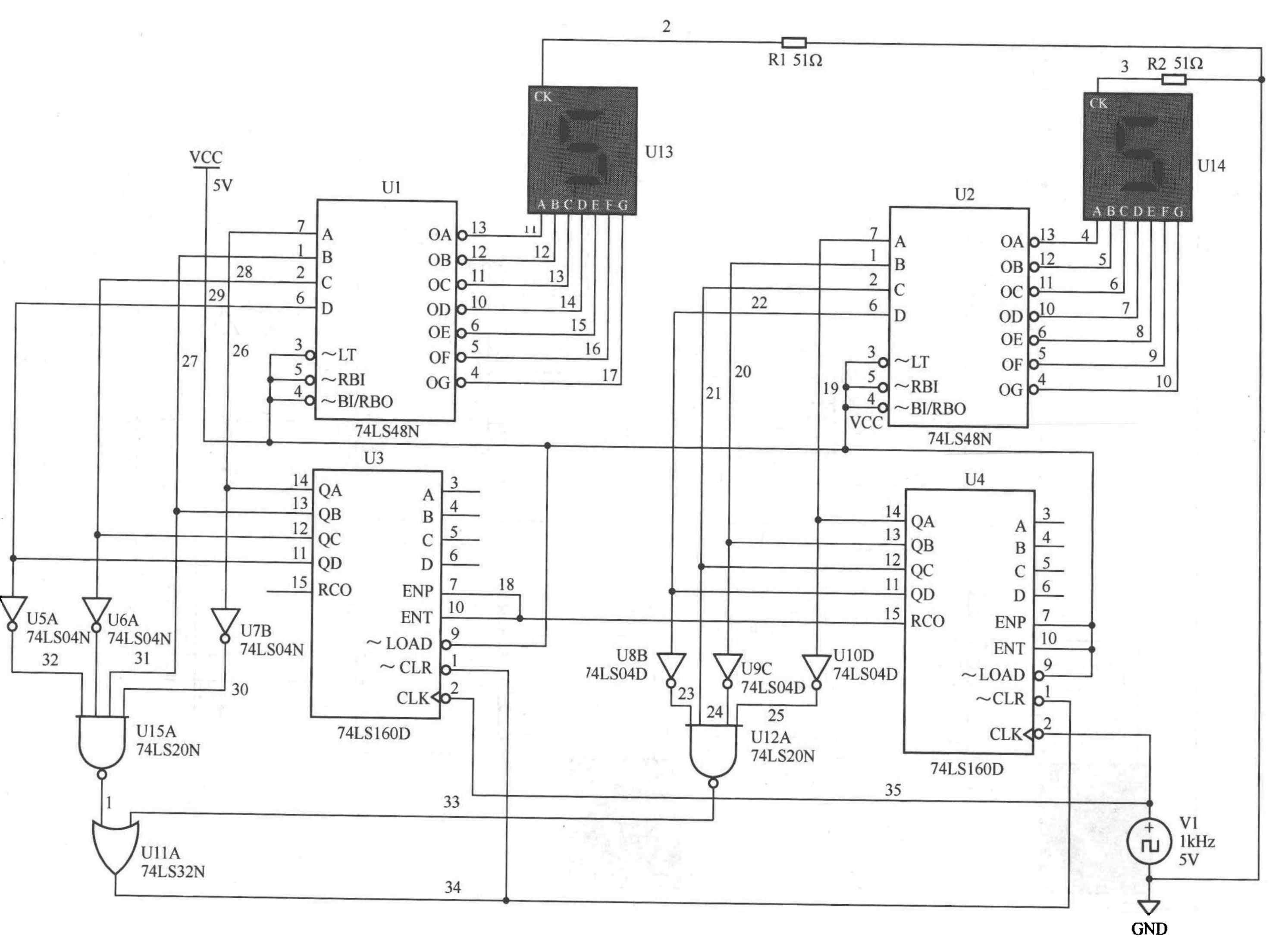


图 2-33 小时计数的二十四进制计数器仿真电路图

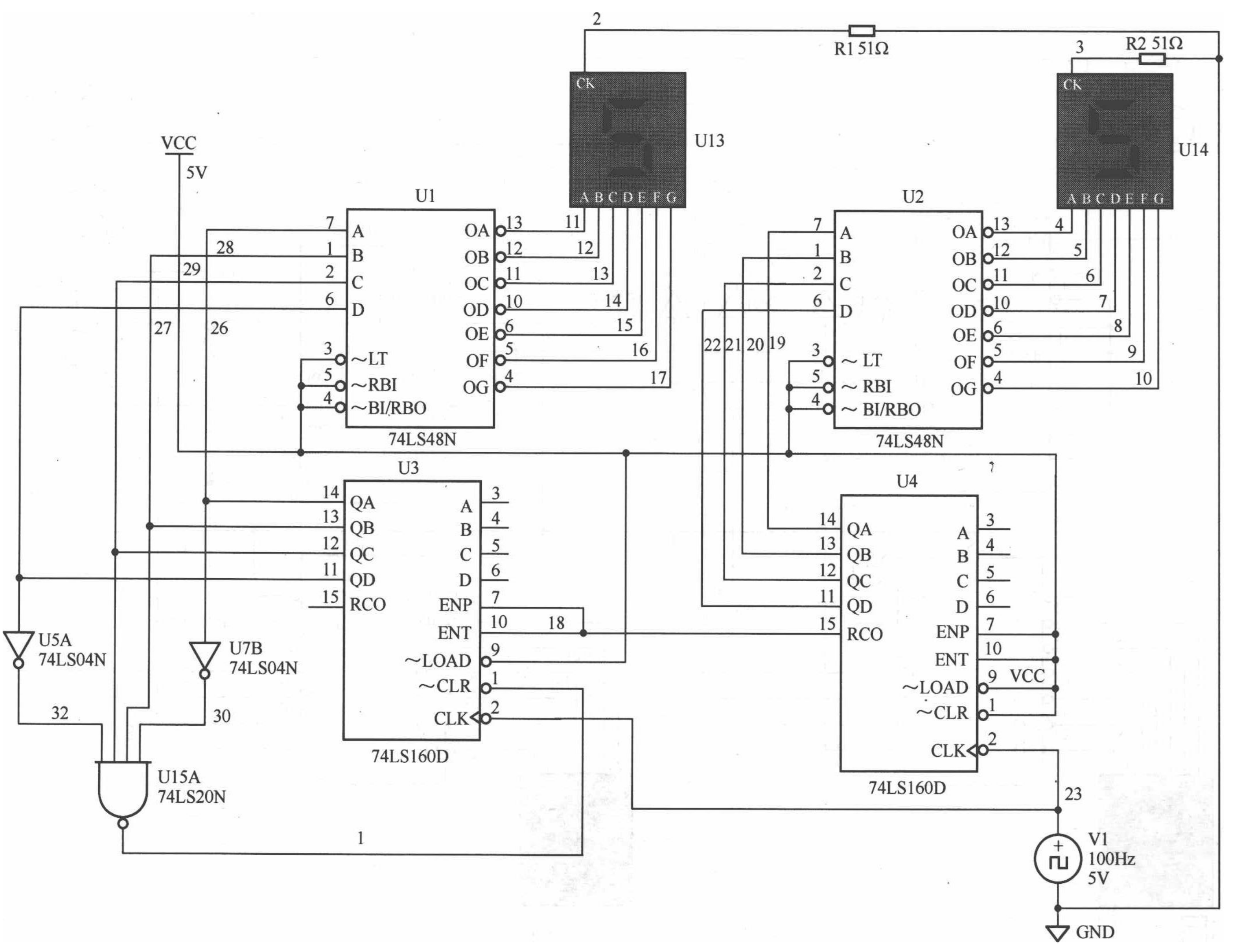


图 2-34　分钟计数的六十进制计数器仿真电路图

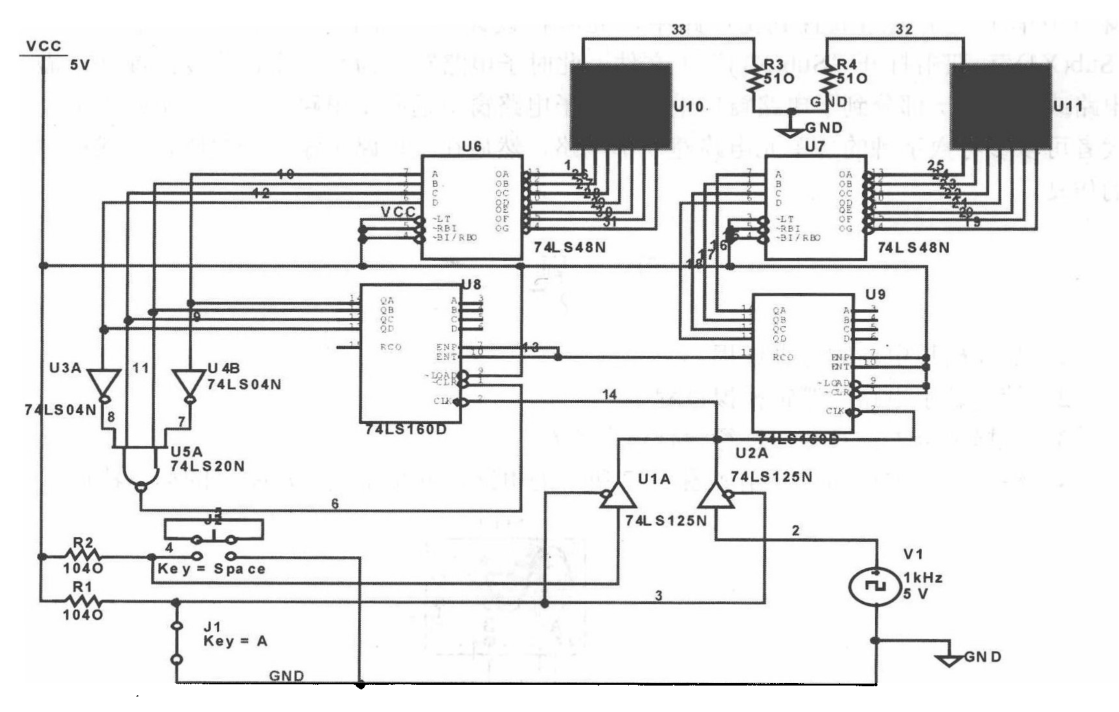


图 2-35　分钟校时控制仿真电路

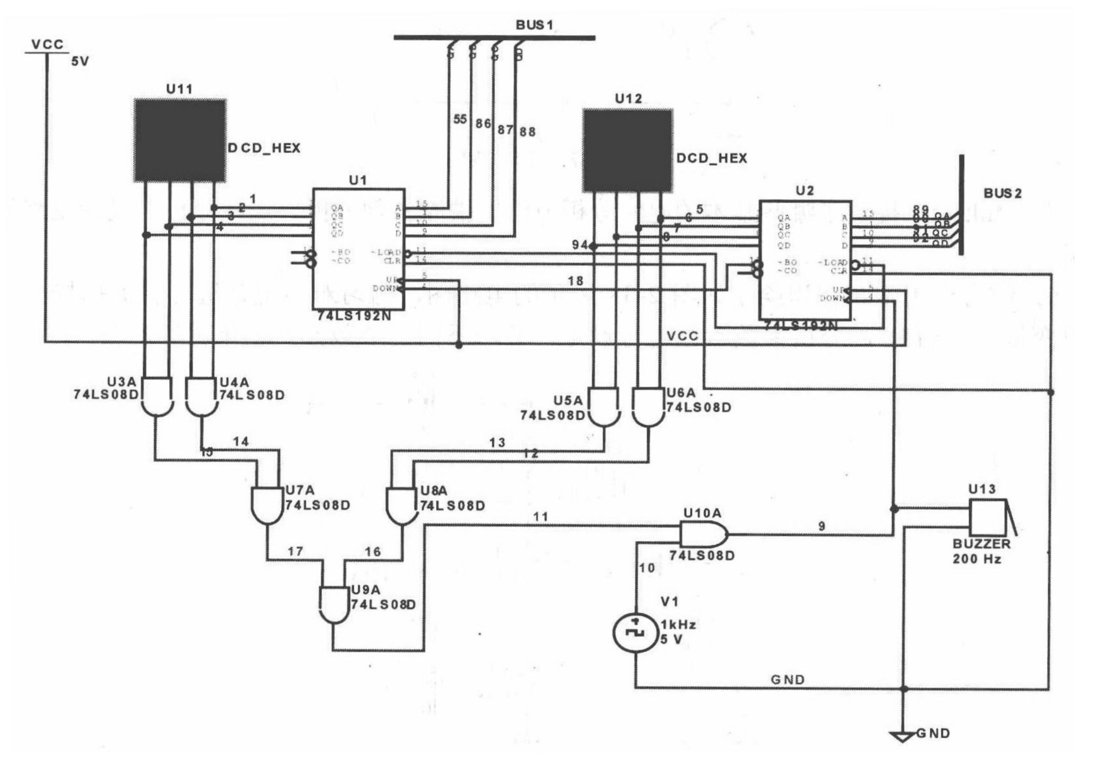


图 2-36　整点报时仿真电路

窗口中单击将子电路放置到主电路中，此时在设计工具箱中主电路文件下加入了子文件“Sub(X1)”。双击打开“Sub(X1)”子文件，此时子电路窗口为空白窗口；复制或剪切需要的电路或电路的一部分到子电路窗口中；关闭子电路窗口返回主电路窗口，子电路添加完成。读者可以参考数字钟的各单元电路建立子电路，然后在主电路中添加子电路，完成整体电路的仿真。

习 题 二

1. 简述模拟电路的仿真过程。
2. 简述数字电路的逻辑模拟过程。
3. 数模混合电路是如何进行混合仿真的？
4. 综合运用本章知识绘制如图 2-37 所示的电路，并用示波器观察它的动态特性。

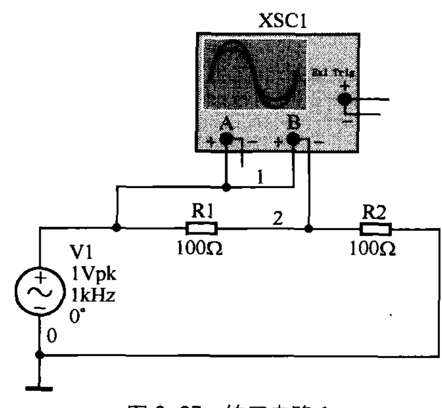


图 2-37 练习电路 1

5. Multisim 提供了哪些基本的电路分析方法？揣摩各种分析方法在分析与设计电路时的作用。

6. 综合运用本章知识绘制如图 2-38 所示的电路图，练习对该电路进行直流扫描分析，输出变量为“V[5]”，分析参数自己进行设定，并对不同的参数分析结果进行比较。

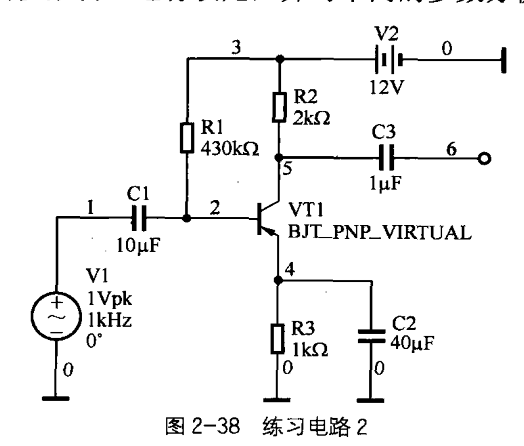


图 2-38 练习电路 2

第3章 印制电路板设计及应用

本章要点

- 印制电路板的抗干扰设计原则
- 布局布线技术
- Protel 99 SE 原理图设计技术
- Protel 99 SE 印制电路板设计技术

3.1 印制电路板基本知识

印制电路板（印刷电路板，PCB 或 PWB）是重要的电子部件，也是电子元器件的支撑体和电子元器件电气连接的提供者。

印制电路板的设计是以电路原理图为根据，实现电路设计者所需要的功能。印制电路板的设计主要指版图设计，需要考虑外部连接的布局、内部电子元件的优化布局、金属连线和通孔的优化布局、电磁保护、热耗散等因素。优秀的版图设计可以节约生产成本，达到良好的电路性能和散热性能。简单的版图设计可以用手工实现，复杂的版图设计则需要借助计算机辅助设计（CAD）实现。

印制电路板的抗干扰设计与具体电路有着密切的关系，这里仅就抗干扰设计的几项常用措施做一些说明。

1．电源线设计

根据印制电路板电流的大小，尽量加粗电源线宽度，减少环路电阻。同时使电源线、地线的走向和数据传递的方向一致，这样有助于增强抗噪声能力。

2．地线设计

地线设计的原则如下。

（1）数字地与模拟地分开。若电路板上既有逻辑电路又有模拟电路，应使它们的地线尽量分开。低频电路的地线应尽量采用单点并联接地，实际布线有困难时可部分串联后再并接地。高频电路宜采用多点串联接地，地线应短而粗，高频元件周围尽量用栅格状大面

积地箔。

（2）接地线应尽量加粗。若接地线用很细的线条，则接地电位随电流的变化而变化，使抗噪性能降低。因此应将接地线加粗，使它能通过三倍于印制板上的允许电流。如有可能，接地线应在 2～3mm 以上。

（3）接地线构成闭环路。设计只由数字电路组成的印制板，其接地电路布成环路大多能提高抗噪声能力。

3．退耦电容配置

PCB 设计的常规做法之一是在印制板的各个关键部位配置适当的退耦电容。

加在芯片上的电源通常是干净的直流，由于电源内阻非零和电路板电源接入点到芯片的距离非零而存在电阻。电路中特别是数字电路中的开关元件在开关时瞬间可产生很大电流，导致芯片的供电端电压瞬间下降使之不能正常工作，标准数字电路的上升和下降边缘时间都是 ns 级，要在这么短的时间内电压从低到高或从高到低，芯片中的有源器件会在短时间内从电源输入或输出很大电流，这要求电源提供的能量是很大的。如果在芯片电源端事先安装储能电容可减小电流瞬变带来的影响。选择原则是动态电流 100mA/0.1uF 电容。PCB 上电容的安装点尽量靠近芯片的电源端，PCB 布线地线尽量宽，以免形成环路引入干扰。在 PCB 板的总电源输入端要加较大电容并配小电容可防止宽带的噪音耦合。

PCB 的常见结构可以分为单层板（single Layer PCB）、双层板（Double Layer PCB）和多层板（Multi Layer PCB）3 种。

单层板（single Layer PCB）是一面敷铜，另一面没有敷铜的电路板。元器件一般情况是放置在没有敷铜的一面，敷铜的一面用于布线和元器件焊接。

双层板（Double Layer PCB）是一种双面敷铜的电路板，两个敷铜层通常被称为顶层（Top Layer）和底层（Bottom Layer），两个敷铜面都可以布线，顶层一般为放置元件面，底层一般为元件焊接面。

多层板（Multi Layer PCB）就是包括多个工作层面的电路板，除了有顶层（Top Layer）和底层（Bottom Layer）之外还有中间层，顶层和底层与双层面板一样，中间层可以是导线层、信号层、电源层或接地层，层与层之间是相互绝缘的，层与层之间的连接往往是通过埋孔来实现的。

采用印制板的主要优点如下。

（1）由于图形具有重复性（再现性）和一致性，减少了布线和装配的差错，节省了设备的维修、调试和检查时间。

（2）设计上可以标准化，利于互换。

（3）布线密度高，体积小，重量轻，利于电子设备的小型化。

（4）利于机械化、自动化生产，提高了劳动生产率并降低了电子设备的造价。

3.2 布局布线技术

PCB 自动布局布线技术是通过计算机软件自动设计组件的摆放位置并将电路原理图中元件的逻辑连接转换为 PCB 铜箔连接的技术。PCB 的自动化设计实际上是一种半自动化的

设计过程，还需要人的参与才设计出合格的PCB。

3.2.1 PCB自动布线技术的步骤

PCB自动布线技术一般遵循以下步骤。

（1）绘制电路原理图。

（2）生成网络表。

（3）建立PCB文件，定义电路板。

（4）加载PCB元件库。

（5）加载网络表。

（6）元件的布局。

（7）设计规则设置与自动布线。

（8）人工布线调整。

（9）PCB电气规则检查及标注文字调整。

（10）图形输出。

3.2.2 元件的布局技术

把元件装入电路板之后，会发现所有的元件重叠在一起，这就需要在所定义的电路板内对元件进行合理的布局。在布局过程中，必须考虑导线的布通率、散热、抗电磁干扰、信号完整性等问题。布局的好坏会直接影响电路板的布线效果及相应电子设备的工作性能，所以合理的布局是PCB设计成功的第一步。一般元件的布局采用自动布局和人工调整（即手动布局）相结合的方法。

Protel 99 SE提供了强大的PCB自动布局功能，PCB编辑器根据一套智能的算法可以自动地将元件分开，然后放置到规划好的布局区域内并进行合理的布局。元件的手动布局是指手动设置元件的位置。自动布局只是对元件进行了初步摆放，元件的摆放并不整齐，需要走线的长度也不是最小，随后的PCB布线效果不会很好，因此需要进行元件的手动布局做进一步调整。

在进行自动布局前，必须在Keep Out Layer上先定义电路板的电气边界，且将当前坐标原点恢复为绝对原点，再加载网络表，否则屏幕会提示错误信息。

关于自动布局和手动布局的细节操作会在3.5.7小节中做详细介绍。

3.2.3 元件的布线技术

完成元件的布局工作后，就可以进入布线操作了。在电路原理图复杂的情况下，如果使用人工布线，不仅效率很低，而且难度也很大，这是可以充分利用Protel 99 SE强大的自动布线功能，快速有效的完成布线工作。

自动布线是指系统根据设计者设定的布线规则，依照网络表中各个元件之间的连线关系，按照一定的算法自动地在各个元件之间进行布线。在PCB上走线的首要任务就是要在PCB上走通所有的导线，这在高密度的PCB设计中很具有挑战性。在能够完成所有走线的前提下，布线的要求如下。

（1）走线长度尽量短而直，在这样的走线上电信号完整性较好。

（2）走线中尽量少的使用过孔。

（3）走线的宽度要尽量宽。

（4）输入输出端的边线应避免相邻平行，以免产生反射干扰，必要时应该加底线隔离。

（5）两相邻层间的布线要互相垂直，平行则容易产生耦合。

自动布线是一个优秀的电路设计辅助软件所必须的功能之一。对于散热、电磁干扰及高频等要求较低的大型电路设计来说，采用自动布线操作可以大大的降低布线的工作量，同时，还能减少布线时的遗漏。

Protel 99 SE 的自动布线功能能够自动分析当前的 PCB 文件，并选择最佳布线方式，但自动布线也会出现一些不合理的布线情况，如有较多的绕线、走线不美观等。此时可以通过手动布线进行调整，对于元件网络较少的 PCB 也可以完全采用手动布线。手动布线，要靠用户自己规划元件布局和走线路径，而网格是用户在空间和尺寸上的重要依据。因此，合理地设置网格，会更加方便设计者规划布局和放置导线。

关于自动布线与手动布线的细节操作会在 3.5.8 小节中做详细介绍。

3.3 Protel 99SE 概述

3.3.1 Protel 99SE 的发展与演变

早在 1987 年和 1988 年，美国的 ACCEL Technologies Inc 公司就推出了第一个应用于电子线路的设计软件包——TANGO，这个软件包开创了电子设计自动化（EDA）的先河，它也可以说是 Protel 的前身。在随后的几年，电子工业的飞速发展使 TANGO 软件包再难以胜任，Protel Technology 公司及时地推出了 Protel for DOS 软件作为 TANGO 的升级版本。

进入 20 世纪 90 年代以来，随着计算机技术的迅猛发展，硬件的整体性能和其相应的软件领域也随之日新月异地进步。于是 Protel Technology 公司在 1991 年推出 Protel for Windows 1.0，这是世界上第一个基于 Windows 操作系统的 PCB（印制电路板）设计工具。随后，Protel Technology 公司又陆续推出了 Protel for Windows 2.0、Protel for Windows 3.0、Protel 98 等产品，直到现在的较新版本以及其改进版 Protel 99SE，之后还出现了 Altium 系列升级版本。由于 Protel 99 SE 版本本身已经比较成熟、界面友好，最重要的是使用它的人群量大等因素，所以本书选择了 Protel 99 SE 这个版本。

3.3.2 Protel 99SE 的设计组件

Protel 99SE 是由众多的服务器程序组成，常用的是以下 5 个组件，分别为原理图设计组件、印制电路板图设计组件、自动布线组件、可编程逻辑设计组件和仿真组件。下面简单介绍原理图设计组件、印制电路板图设计组件和自动布线组件。

1. 原理图设计组件

基于 Windows 平台的 Protel 99SE 中的原理图设计组件是一个功能完备的多图样、层次化的原理图编辑器。它包含了众多的高级设计工具，可高效地实现电子产品的原理图设计。

原理图设计组件的编辑器具有丰富而又强大的编辑功能，使用的是交互式全局编辑。电气栅格特性提供了所有电子器件的自动连接，这使得手工连线变得更为简捷。

原理图设计组件还具有强大的电气检查功能，能够快捷的对大型的复杂电路进行检查，并可将检查结果直接标记在原理图中，方便了原理图的修正工作。原理图设计组件具备完善的库元件编辑和管理功能。其中原理图设计器提供超过六万个元件的元件库，如果用户从这些库中没有找到合乎自己要求的元件，还可以自行创建原理图元件。

2. 印制电路板图设计组件

印制电路板设计系统主要用于电子产品的电路板设计，完成整个电子产品设计过程中物理结构的设计，包括印制电路板的机械结构设计、元件的布局设计和电路的布线设计。设计的结果可以用光绘数据文件的形式输出。

与原理图编辑器一样，印制电路板图编辑器也具备了丰富而又灵活的编辑功能，提供交互式的全局编辑，对象属性的修改操作和原理图中完全一样。印制电路板图设计组件借助于自动布线组件，能够实现设计的自动化，同时具备设计规则检查，以修正违反设计规则的错误。印制电路板图设计组件提供了较为齐全的封装元件，具备了完善的封装元件库管理功能，用户可以方便快捷地创建新的封装元件。

印制电路板图设计组件还具备良好的兼容性。其编辑器可以调入众多 EDA 设计系统的设计文件，如 PADS、OrCAD、TANGO、Protel for DOS 等版本的设计文件。

3. 自动布线组件

Protel 99SE 中的自动布线组件是一个完全集成的基于形状的无网络自动布线器，采用基于人工智能的布局布线方法。它使用方便、布线效率高，可以对印制电路板板面进行优化，实现高难度、高精度的印制电路板图自动布线，布线质量能达到专业级水平。该组件主要是为印制电路板图设计组件而服务的，以实现电路设计自动化。在 Protel 99SE 中，该组件没有独立的用户接口界面，而纯粹作为一个内嵌的服务器程序，通过印制电路板图编辑器来实现与用户的交互。

3.4 用 Protel 99SE 设计原理图

3.4.1 原理图设计过程

原理图设计过程如下。

（1）设计图纸大小。进入 Protel 99/Schematic 设计环境后，首先要构思好元件图，设计好图纸大小。图纸大小是根据电路图的规模和复杂程度而定的，设置合适的图纸大小是设计好原理图的第一步。

（2）设置 Protel 99/Schematic 设计环境。设置 Protel 99/Schematic 设计环境，包括设置格点大小和类型，光标类型等，大多数参数也可以设置系统默认值。

（3）放置元件。用户根据电路图的需要，将元件从元件库里取出放置到图纸上，并对放置元件的序号、元件封装进行定义和设定等工作。

（4）原理图布线。利用 Protel99/Schematic 提供的各种工具，将图纸上的元件用具有电器意义的导线、符号连接起来，构成一个完整的原理图。

（5）调整线路。将初步绘制好的电路图做进一步的调整和修改，使得原理图更加美观。

（6）报表输出。通过 Protel99/Schematic 提供的各种报表工具生成各种报表，其中最重要的报表是网络表，通过网络表为后续的电路板设计做准备。

下面结合图 3-1 所示实例来说明原理图的设计过程。

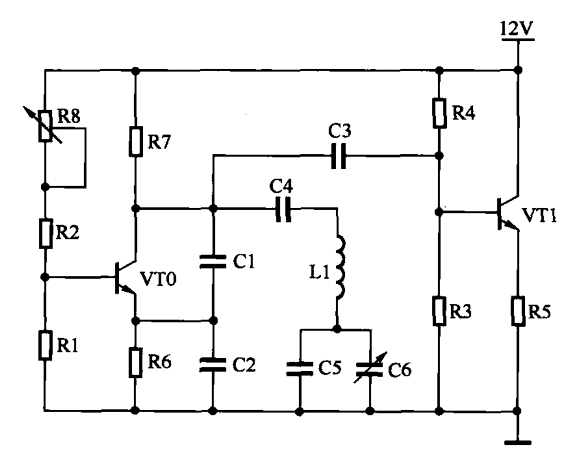


图 3-1　电容三点式振荡电路

3.4.2　新建一个设计库

新建设计库的步骤如下。

（1）启动 Protel 99，出现启动界面，如图 3-2 所示。启动后的窗口如图 3-3 所示。



图 3-2　启动界面

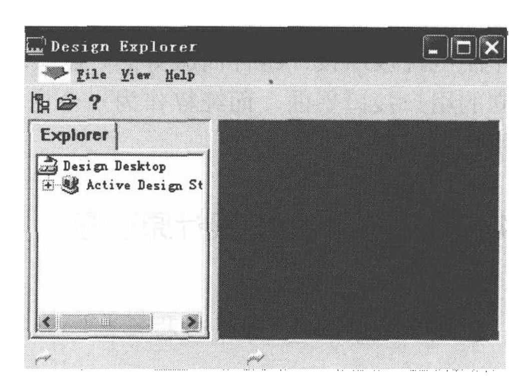


图 3-3　启动后的窗口

（2）单击“File/New”菜单命令来新建一个设计库，出现如图 3-4 所示对话框。在“DataBase File Name”处输入设计库存盘文件名“circuit.ddb”，单击“Browse…”按钮可改变存盘目录。

如果想用口令保护设计文件，可以单击 Password 选项卡，再选 Yes 并输入口令，单击“OK”按钮后，进入如图 3-5 所示的主设计窗口。

（3）单击“File/New…”菜单命令，打开 New Document 对话框，如图 3-6 所示，单击“Schematic Document”菜单命令，建立一个新的原理图文档。

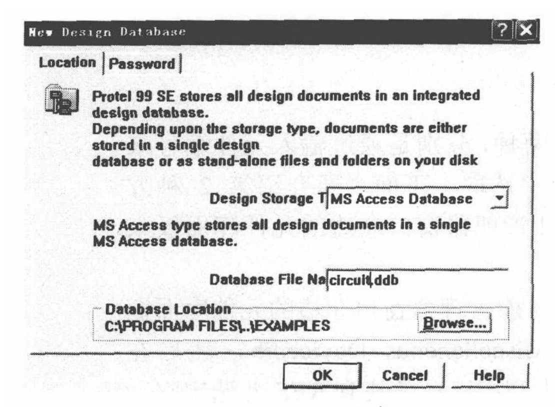


图 3-4 新建设计库对话框

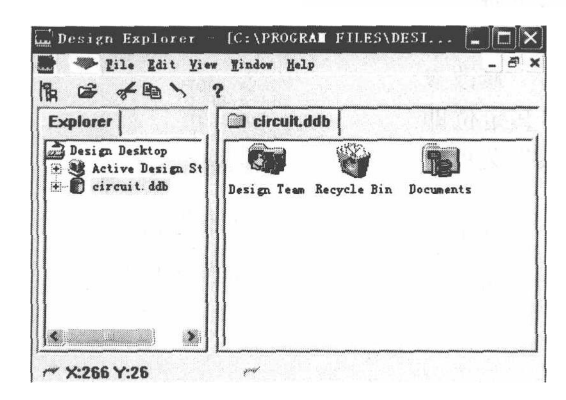


图 3-5 主设计窗口

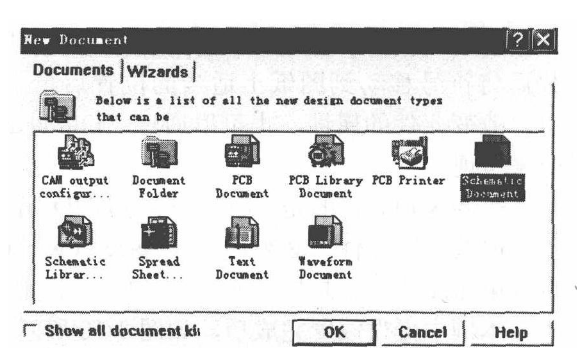


图 3-6 新建文档对话框

3.4.3 设置图纸大小和添加元件库

在放置元件之前，必须先将该元件所在的元件库加载到设计环境才行，添加元件库的步骤如下。

（1）双击设计管理器中的 Sheet 1.Sch 原理图文档图标，打开原理图编辑器。

（2）单击“Design/Options”菜单命令，设置图纸大小为 A4，其余默认。

（3）单击设计管理器里的 Browse Sch 选项卡，然后单击“Add/Remove”按钮，屏幕将出现如图 3-7 所示的“元件库添加、删除”对话框。

（4）在“Design Explorer 99/Library/Sch”文件夹下选取元件库文件，然后双击或单击“Add”按钮，此元件库就会出现在 Selected Files 框中。

（5）然后单击“OK”按钮，完成该元件

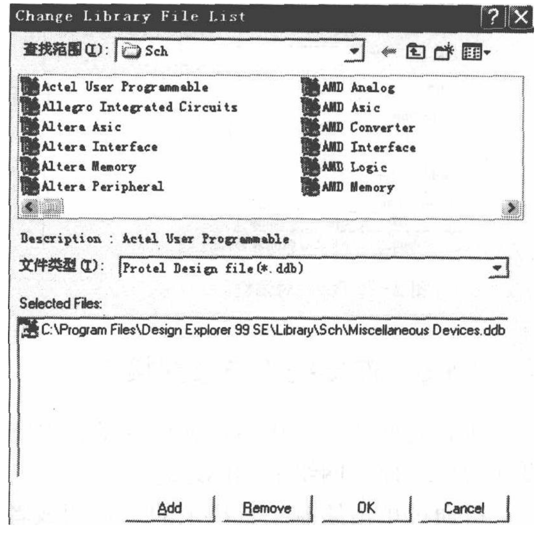


图 3-7 “元件添加、删除”对话框

库的添加。

3.4.4　放置元件

选取元件的方法有两种，分别是通过输入元件编号来选取元件和从元件列表中选取。下面主要介绍第 2 种方法。该操作必须通过设计管理器窗口左边的元件库面板来进行。

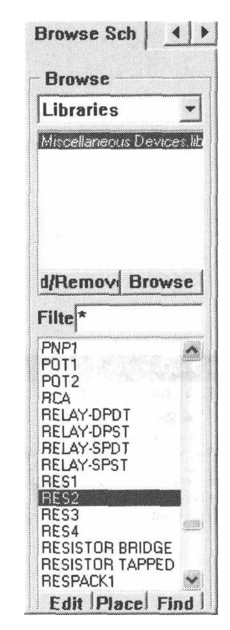


图 3-8　选取元件

如放置电阻，在设计库管理器窗口左边的元件库面板的 Library 栏中选择 Miscellaneous Device.lib，然后在 Components In Library 中利用滚动条找到 RES 并选定它，如图 3-8 所示。然后，单击“Place”按钮，此时屏幕上会出现一个随鼠标移动的 RES 符号，按空格键可旋转元件，按<Tab>键可打开编辑元件对话框，如图 3-9 所示。修改参数后将符号移动到图纸上适当的位置后单击将其定位即可。改变元件的属性，也可以通过“Edit/change”菜单命令来实现。

需要区别对待的是 VCC 元件与 GND 元件，它们不同于一般的电器元件，必须通过“Place/Power Port”菜单命令或电路图绘制工具栏上的⏚按钮才可调用。

本例中元件放置完成后，如图 3-10 所示。

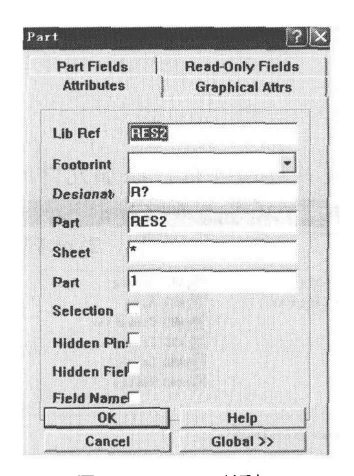


图 3-9　Part 对话框

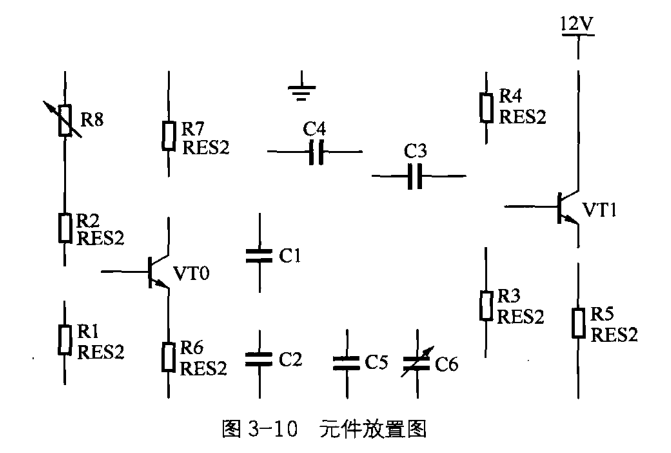


图 3-10　元件放置图

3.4.5　连接线路与放置接点

所有元件放置完毕后，就可以进行电路图中各对象的连线，连线的主要目的是按照电路设计的要求建立网络的实际连通性。

可通过单击绘制工具栏上的≈按钮或者单击“Place/Wire”菜单命令将编辑状态切换到连线模式。此时，鼠标指针由空心箭头变为大十字，只需将鼠标指针指向欲拉连线的元件端点，

单击鼠标，就会出现一条随鼠标指针移动的预拉线，当鼠标的指针移动到连线的转弯点时，单击鼠标就可以定位一次转弯。当拖动虚线到元件的引脚上并单击，就会终止此次连线。若想将编辑状态回到待命模式，可右击或按下<Esc>键。

要放置接点，可单击电路绘制工具栏上的按钮或单击“Place/Junction”菜单命令，此时鼠标指针 building 由空心箭头变成十字，且还有一个小黑点，将鼠标指针指向欲放置接点的位置，单击鼠标即可，右击或按下<Esc>键退出连接接点状态。

在连线或放置器件时，要删除不需要的导线、节点或元件，可以单击“Edit/Delete”菜单命令，然后就可以在原理图界面删除不需要的部件了。退出删除命令，可以右击或按下<Esc>键。

布线完成后的原理图如图 3-1 所示。

3.4.6　电气规则检查

单击“Tools/ERC”菜单命令，对画好的电路原理图进行电气规则检查，检查完毕后，屏幕显示 ERC 报告文件，如图 3-11 所示，如没有错误，就可以进入下一步；如有错误就根据错误提示修改原理图直到通过 ERC 检查，如图 3-12 所示。

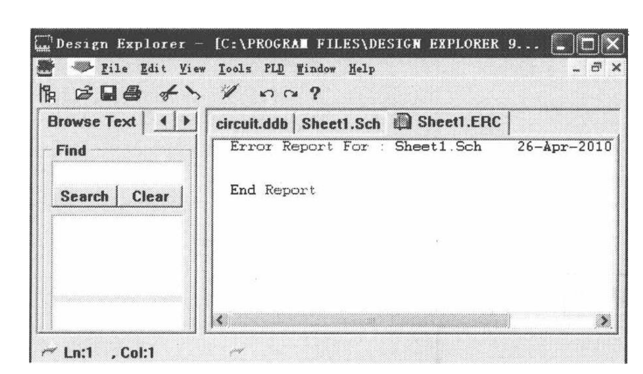


图 3-11　ERC 报告文件

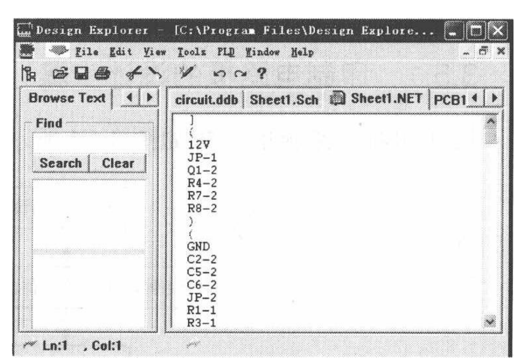


图 3-12　网络表文件

3.4.7　建立网络表

设计电路原理图的目的是进行 PCB 设计，网络表则是联系电路原理图和 PCB 的桥梁。所谓网络，指的是彼此连接在一起的一组元件引脚。一个电路实际上就是由若干网络组成的，而网络报表就是对电路或者电路原理图的一个完整描述。描述的内容包括两个方面：一是电路原理图中所有元件的信息（包括元件标识、元件引脚、PCB 封装形式等）；二是网络的连接信息（包括网络名称、网络节点等），它们是进行 PCB 布线，设计 PCB 不可缺少的工具。

网络报表有多种形式，通常为一个 ASCII 码的文本文件，网络报表用于记录电路中各个文件的数据和描述各个元件之间的连接关系。在以往低版本的设计中，往往需要生成网络报表以便进行下一步的 PCB 设计或进行仿真。Protel 99 SE 提供了集成的开发环境，用户不用生成网络报表就可以直接生成 PCB 或进行仿真。后面介绍的例子就是直接生成 PCB 的。但有时为方便交流与查错，还是要生成网络报表。

网络报表的生成有多种方法，可以在原理图编辑器中由电路原理图文件直接生成，也可

以利用文本编辑器手动生成，当然，还可以在 PCB 编辑器中，从已经布线的 PCB 文件中导出相应的网络报表。下面将介绍第一种方法，这也是最常用的一种方法。

单击“Design/Create Netlist”菜单命令，建立如图 3-12 所示的网络表。可以看到，随后会出现网络表文件“*.net”。

3.4.8 保存文件

电路图绘制完成后要保存起来，以供日后调出修改及使用。当打开一个旧的电路图文件并进行修改后，单击“File/Save”菜单命令可自动按原文件名将其保存，同时覆盖原先的旧文件。在保存文件时如果不希望覆盖原先的文件，可以换名保存。具体方法是单击“File/Save As…”菜单命令，打开如图 3-13 所示对话框，在对话框中指定新的存盘文件名就可以了。

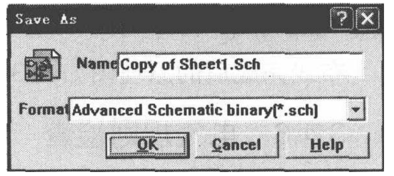


图 3-13　换名存盘对话框

3.5 用 Protel 99SE 设计印制电路板

3.5.1 印制电路板的设计步骤

设计印制电路板的步骤如图 3-14 所示。

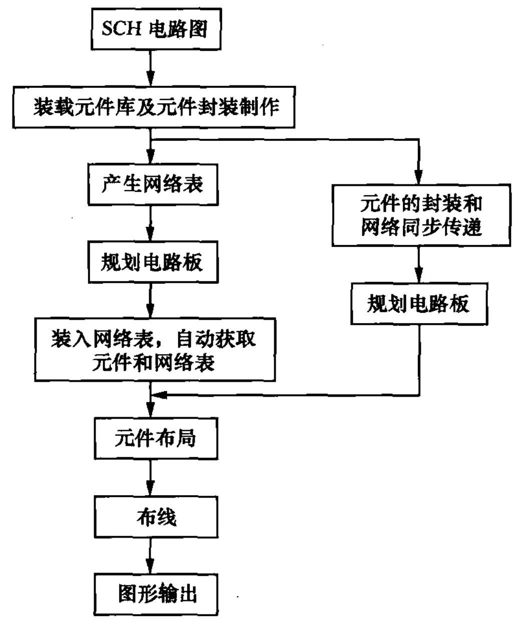


图 3-14　设计印刷电路板的流程图

3.5.2 创建 PCB 图文件

新建一个 PCB 文件可以进入设计文件夹“Document”，单击“File/New”菜单命令或在

工作区右击选择 New 选项，会弹出如图 3-15 所示的选择文件类型的对话框。

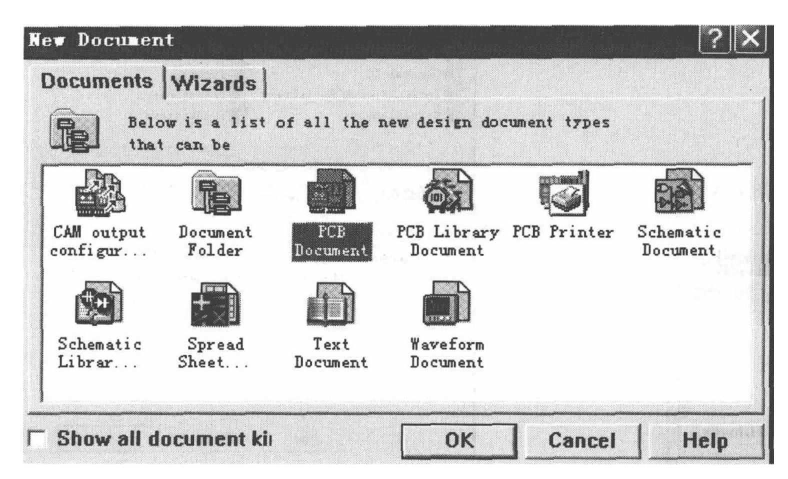


图 3-15　选择文件类型对话框

双击该对话框中的“PCB Document”图标，即可创建一个新的印制板电路图文件，默认的文件名为“PCB 1.PCB”。在工作窗口中该文件的图标上单击或在设计浏览器中该文件的文件名上双击，即可进入图 3-16 所示的印制电路板编辑器。

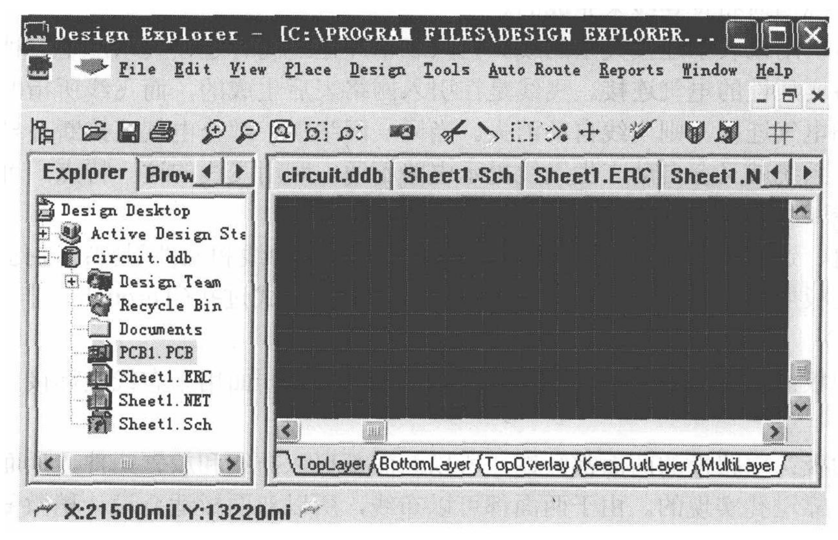


图 3-16　PCB 图编辑器

3.5.3　装载元件库

在浏览器的组合框中选择库“Libraries”，如图 3-17 所示。单击“Add/Remove”按钮，将出现如图 3-18 所示的关于引入库文件的对话框。

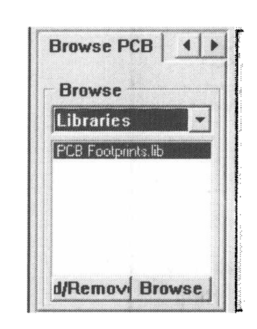


图 3-17 装载元件库

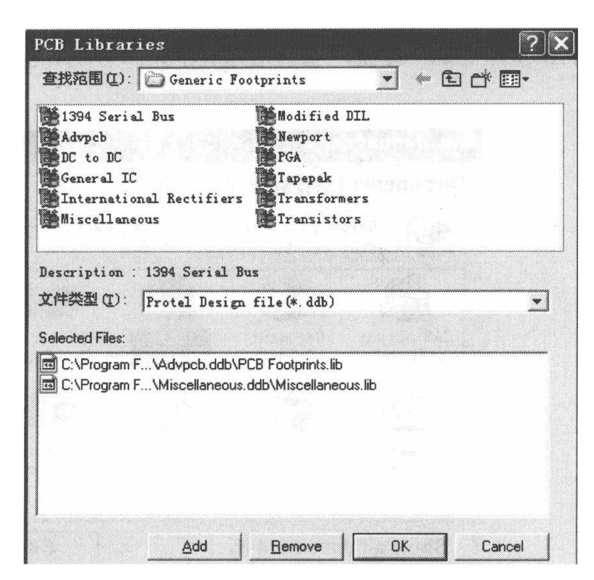


图 3-18 元件库管理对话框

3.5.4 设置电路板工作层面

1. 有关电路板的几个基本概念

（1）铜膜线：简称导线，是敷铜静腐蚀后形成的用于连接各个焊点的导线。印制电路板的设计都是围绕如何布置导线来完成的。

（2）飞线：用来表示连接关系的线。它只表示焊盘之间有连接关系，是一种形式上的连接，并不具备实质性的电气连接。飞线是在引入网络表后生成的，而飞线所指的焊盘间一旦完成实质性的电气连接，则飞线自动消失。当同一网络中，部分电气连接断开导致网络不能完全联通时，系统就又会自动产生飞线提示电路不通。利用飞线的这一特点，可以根据电路板中有无飞线来大致判断电路板是否已经完成布线。

（3）焊盘、过孔：焊盘（Pad）的作用是放置、连接导线和元件引脚；过孔（Via）的作用是实现不同板层间的电气连接。过孔主要分为 3 种：穿透式过孔（Through）、半盲孔（Blind）、盲孔（Buried）。

（4）单面板：电路板一面敷铜，一面不敷铜，敷铜的一面用来布线及焊接，另一面用来放置元件。

（5）双面板：电路板的两面都敷铜，所以两面都可以布线和放置元件，顶面和底面之间的电气连接是靠过孔实现的。由于两面都可以布线，所以双面板适合设计稍微复杂的电路，应用较广泛。

长度单位及其换算：Protel 99SE 的 PCB 编辑器支持英制（mil）和公制（mm）两种长度单位，它们的换算关系是：100mils=2.54mm。单击“View/Toggle Units”菜单命令或按快捷键<Q>就能实现这两种长度单位之间的转换。

安全间距：进行印刷电路板的设计时，为了避免导线、过孔焊点及元件的相互干扰，必须使它们之间留出一定的距离，这个距离称为安全间距（Clearance）。

2．工作层面的类型

Protel 99SE 提供了若干不同类型的工作层面，包括信号层（Signal layers)、内部电源/接地层（Internal plane Layers)、机械层（Mechanical Layers)、阻焊层（Solder mask Layers)、锡膏防护层（Paste mask Layers)、丝印层（Silkscreen Layers)、钻孔位置层（Drills Layers）和其他工作层面（Others)。

3．设置工作层面

单击“Design/Option”菜单命令，弹出“Document Option”对话框，选择其中的“Layers”选项即可进入工作层面对话框，如图 3-19 所示。进入 Options 选项卡，如图 3-20 所示。在该选项卡中可对 Gird（栅格)、Electrical Gird（电气栅格)、Measurement（计量单位）等选项进行选定。

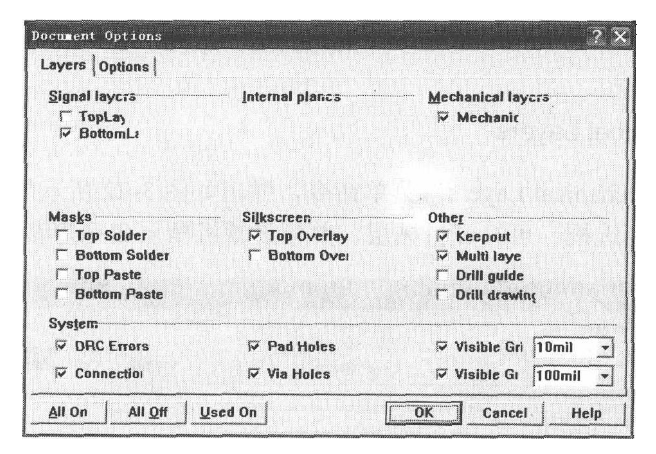


图 3-19　工作层面设置对话框

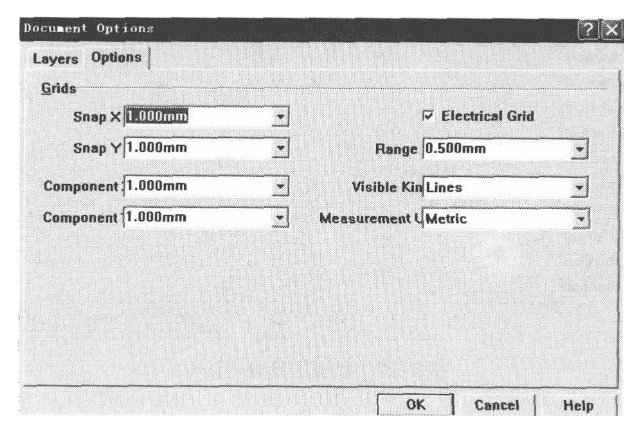


图 3-20　网格、电气栅格及计量单位设置对话框

4．设置信号层与内部电源/接地层

单击“Design/Layer Stack Manager”菜单命令，弹出如图 3-21 所示工作层面管理对话框。

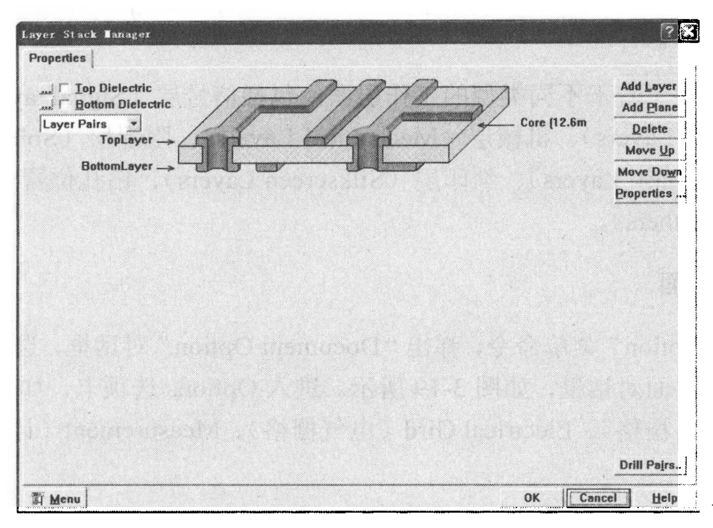


图 3-21　工作层面管理对话框

5. 设置 Mechanical Layers

单击“Design/Mechanical Layers”菜单命令，弹出如图 3-22 所示的机械层设置对话框，单击“Mechanical”复选框，可打开机械层，并可设置机械层名称等参数。

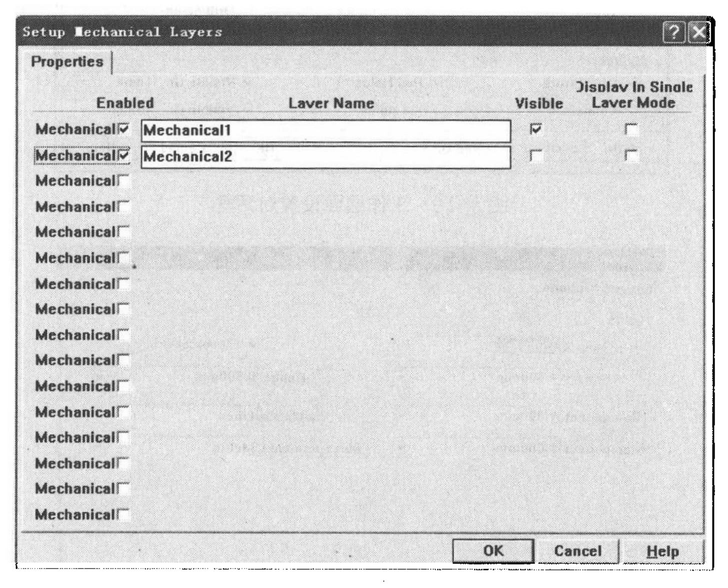


图 3-22　机械层设置对话框

3.5.5　规划电路板

所谓规划电路板，就是根据电路的规模以及公司或制造商的要求具体确定所需制作电路板的外形尺寸和电气边界。电路板规划的原则是在满足公司或制造商的前提下，尽量美观且便于后面的布线工作。

首先选择当前的工作层面为“Keep Out Layer”：单击“KeepOutLayer”按钮即可将当前的工作层面切换到 Keep Out Layer 层面。在该层面上确定电路板的电气边界位置，如图 3-23 所示。

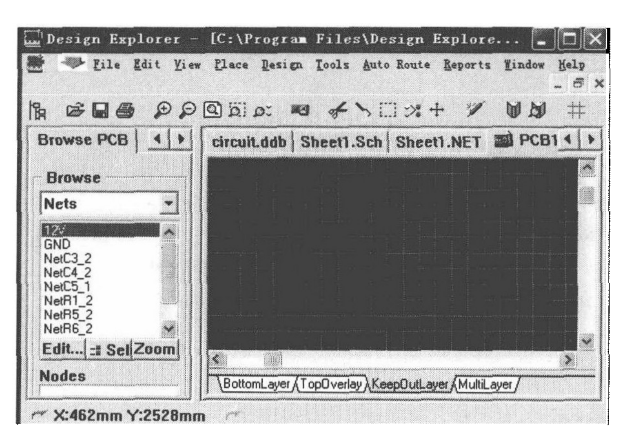


图 3-23　绘制电路图边界

3.5.6　装入网络表与元件

规划好电路图后，接下来就是要装入网络表和元件，网络表和元件是同时装入的。网络表和元件的装入过程，实际上就是将原理图的设计数据装入印制电路板的设计系统 PCB 的过程。下面将介绍两种装入网络表与元件的方法。

1．利用设计同步器装入网络表和元件

在原理图编辑器中单击“Design/Update PCB”菜单命令，弹出如图 3-24 所示的对话框。本例生成的 PCB 图如图 3-25 所示。

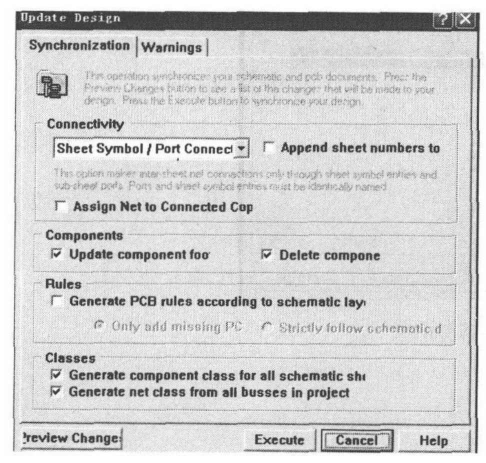


图 3-24　更新 PCB 设计对话框图

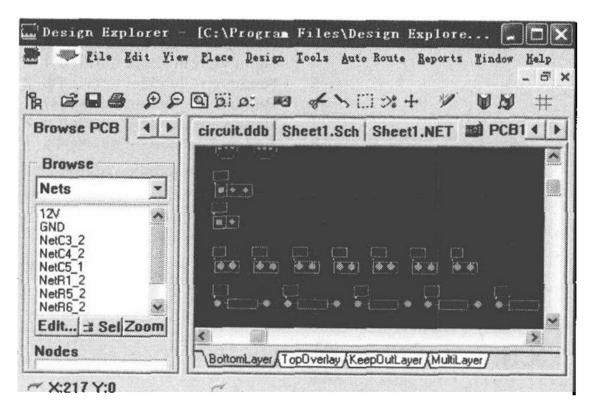


图 3-25　生成的 PCB 电路图

2．利用原理图生成的网络表文件装入网络表和元件

在利用网络表文件装入网络表和元件时，可以在 PCB 编辑器中单击“Design/Load Nets”

菜单命令，弹出如图 3-26 所示的装入网络表的对话框。通过 Browse 找到需要的 Netlist File 即可，注意图 3-27 所示的 Error 处是否有错误，即装载完网络表后要检查网络表是否有错误，若有错误，则返回原理图并更改错误。

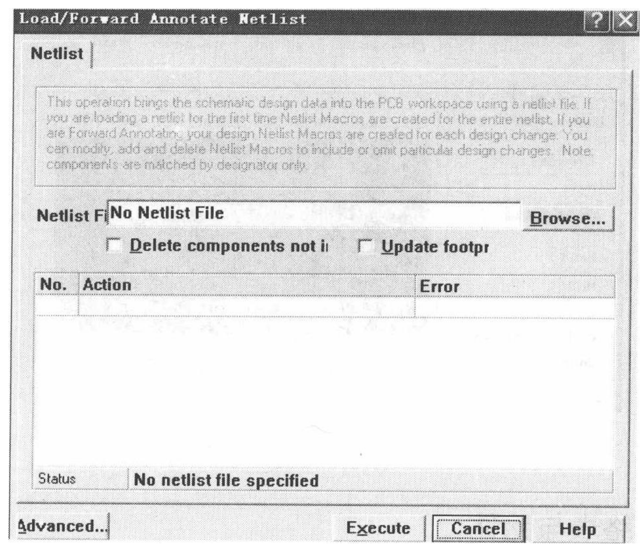


图 3-26　装入网络表对话框

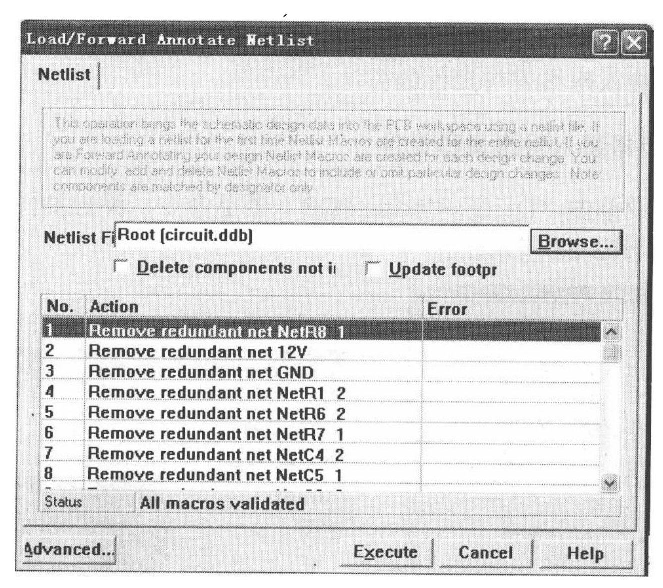


图 3-27　载入网络表后的对话框

3.5.7　元件布局

1．元件的自动布局

Protel 99SE 提供了元件的自动布局功能。通过程序算法自动将元件分开，放置在规划好的电路板电气范围内。元件自动布局的实现方法可以单击“Tools/Auto Placement/Auto

Placer…”菜单命令，弹出如图 3-28 所示对话框。合理设置后即可完成自动布局，由于本例元件不多，直接手动布局即可。图 3-29、图 3-30、图 3-31、图 3-32 所示分别为自动布局前、后、手工调整元件位置、完成调整后的电路图。读者可以在以后的设计实践中尝试一下该软件的自动布局技术。

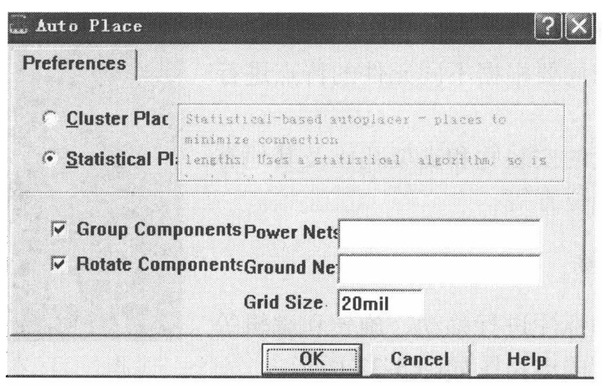


图 3-28　自动布局对话框

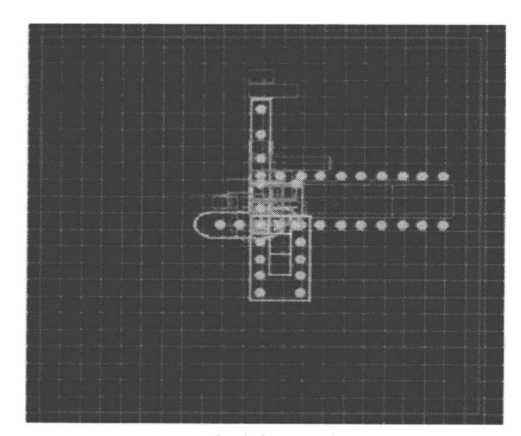


图 3-29　自动布局前的电路图

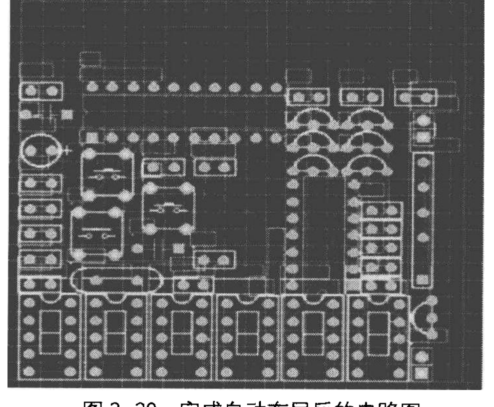


图 3-30　完成自动布局后的电路图

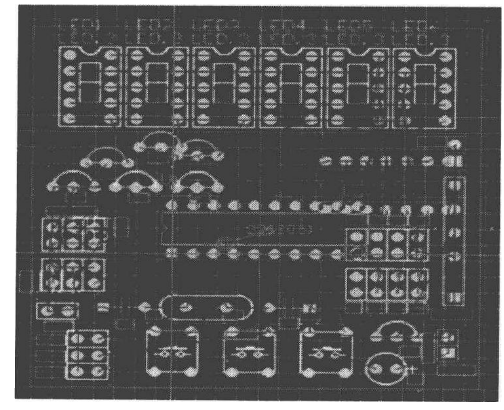


图 3-31　手工调整元件位置

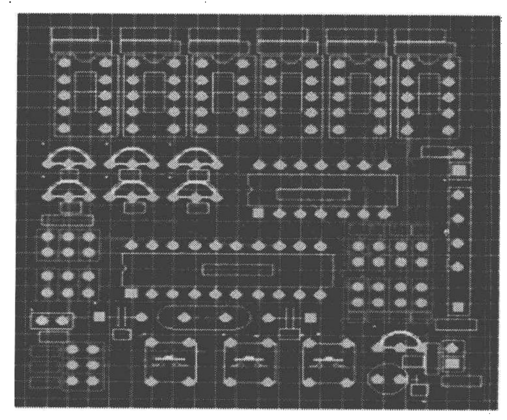


图 3-32　完成调整后的电路图

2．元件的手工布局及调整

元件的布局要考虑以下几个方面的问题。

（1）元件的布局应便于用户的操作使用。

（2）尽量按照电路的功能布局。

（3）数字电路部分与模拟电路部分应尽量分开。

（4）特殊元件的布局要根据不同元件的特点进行合理布局。

（5）应留出电路板的安装孔和支架孔以及其他有特殊要求的元件的安装位置等。

3．元件标注的调整

用户可以对元件的标注进行移动、旋转和编辑等操作。本例手动布局后的电路图如图 3-33 所示。

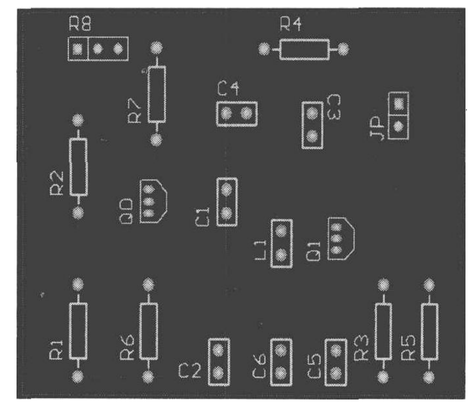


图 3-33　本例手动布局后的电路图

3.5.8　自动布线

自动布线是指 Protel 99SE 程序根据用户设定的有关布线参数和布线规则，按照一定的算法，依照网络表所制定的连接关系自动在各个元件之间进行布线，从而完成印制电路板的布线工作。

1．自动布线规则设置

自动布线前，首先要设置布线规则，设计规则制定后，程序自动监视 PCB，检查 PCB 中的图件是否符合设计规则，若违反了设计规则，将以高亮显示错误内容。

单击“Design/Rules”菜单命令，弹出如图 3-34 所示的对话框，此对话框共有 6 个选项卡，分别设定与布线、制造、高速线路、元件自动布置、信号完整性分析及其他方面有关的

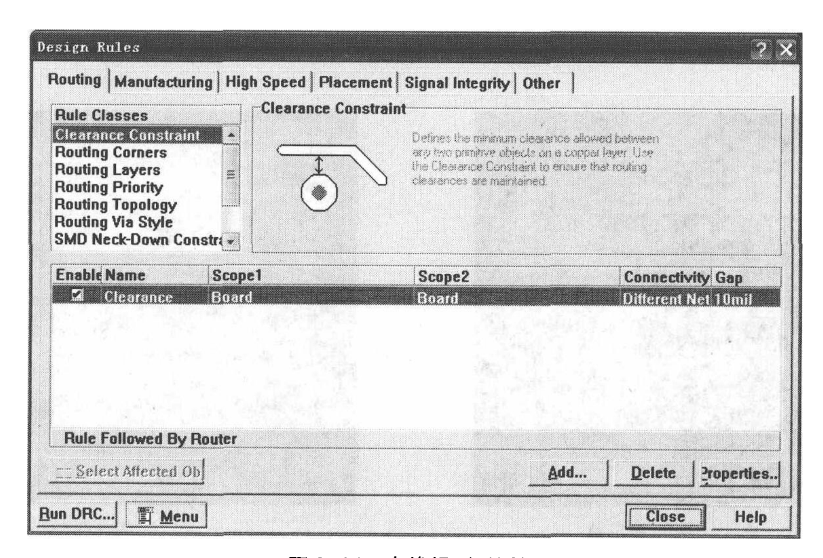


图 3-34　布线规则对话框

设计规则。图中选中的是有关布线的设计规则（Routing），在此选项卡中，Rule Classes 栏中列出了有关布线的 10 个设计规则，右上方区域是所选取的设计规则的说明，下方是所选取的设计规则的具体内容。

（1）布线的层面。Protel 99SE 的 PCB 编辑器，可以在 16 个 Signal Layer（信号层面）上实现布线，每个都可以单独设置布线属性。对双面板的自动布线来说，应该让系统仅在 Toplayer（顶层）和 Bottomlayer（底层）两个层面布线；对单面板则只能在 Bottomlayer（底层）布线。其他的信号层应该处于禁止自动布线状态，否则系统会自动使用任何可用的信号层。

设定层面的布线属性，可在"Design/Rules"对话框中设定。单击"Routing"选项卡，选中 Rule Classes 列表框中的"Routing Layers"项，然后单击"Routing Layers Rule"按钮，弹出的对话框如图 3-35 所示。

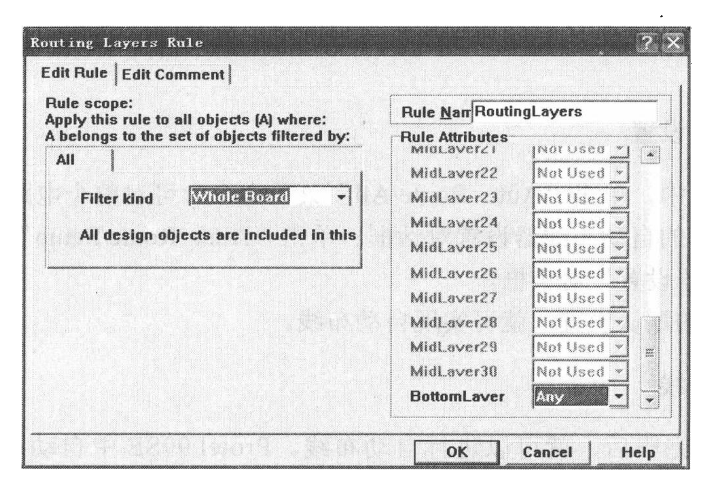


图 3-35 布线层面对话框

对话框中的 Rule Attributes 选项，就是双面板常用的层面布线属性。其中水平布线 Toplayer="Horizontal"，底层设置成垂直布线 Bottomlayer="Vertical"，14 个中间信号层都设置成不使用，即 1～14="Not Used"。制作单面板时，因为只有底层是实际布线层，所以要把 Bottom Layer 的布线方向设定为任意，即 Bottomlayer="Any"，其他层都设置成不使用，即"Not Used"。

（2）布线的其他规则。Protel 为自动布线规定了很多默认的规则，设计者通常不需要重新设置，但是应该了解这些规则的内容和意义，以便需要时对其更改。

① Clearance Constraint（安全距离）：两个导体之间的最小距离，默认值=10mil。

② Routing Corners（布线拐角模式）：默认值=45 度。

③ Routing Priority（布线优先级别）：范围 0～100，数值越大，级别越高；自动布线时，级别高的先走线。

④ Routing Topology（自动布线方式）：默认值设置为"Shortest（最短布线长度）"。

⑤ Routing Via Style（过孔类型）：有 3 个参数："Style（类型）"默认值为"Through（通孔）"，孔径默认值为"Hole Size=28mil"，外径默认值为"Width=50mil"。

⑥ SMD Neck-Down Constraint（SMD 焊盘与导线比例规则）：此规则用于设置 SMD 焊盘在连接导线处的焊盘宽度与导线宽度的比例，可定义一个百分比，默认值是 50%。

⑦ SMD To Corner Constraint（SMD 与拐角最小间距限制）：此规则用于设置 SMD 焊盘与导线拐角的间距大小，默认值是 0mil。

⑧ SMD To Plane Constraint（SMD 焊盘与电源层过孔间的最小长度规则）：此规则用于设置 SMD 焊盘与电源层中过孔间的最短布线长度。默认值是 0mil。

⑨ Width Constraint（布线宽度）：默认值为 10mil。

以上这些规则可在如图 3-34 所示的对话框的 Routing 选项卡 Rule Classes 列表框中找到，每项规则大多可以有多个不同对象设置值。例如，布线宽度可以按网络名分别设置，电源 VCC 和接地 GND 的布线宽度可以设定得粗一些，其他设定的细一些。单击对话框中的“Add…”按钮，可将多个作用对象的设置值，加到“Scope”列表框中。

由于可以设定的布线规则非常多，而且有些参数之间又互相有牵连，设置不合适的话，很可能适得其反。因此，初学者一般使用默认值即可，等对布线规则的理解加深后，再进行高标准的规则设置。

2．自动布线器设置

在 PCB 编辑器中，单击“Auto Route/All”菜单命令，可对整个电路板进行自动布线，弹出如图 3-36 所示的自动布线器设置对话框。单击“Auto Route/Setup”菜单命令，同样也会弹出“自动布线器设置”对话框。

采用对话框中的默认设置，就可实现自动布线。

3．运行自动布线

布线参数设置完毕后，就可以进行自动布线。Protel 99SE 中自动布线的方式有很多，既可以进行全局布线，也可以对用户指定的区域、网络、元件甚至是连接进行布线，用户可以根据需要选择最佳的方式。图 3-37 所示为完成布线的电路图。注意电源线和地线要加粗。

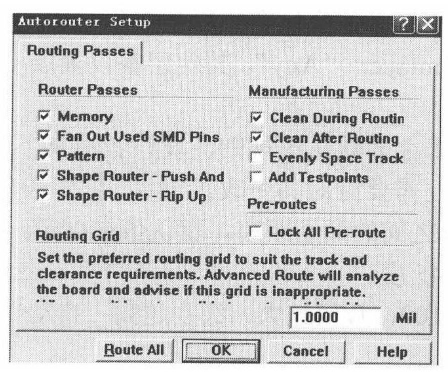


图 3-36 自动布线参数设置对话框

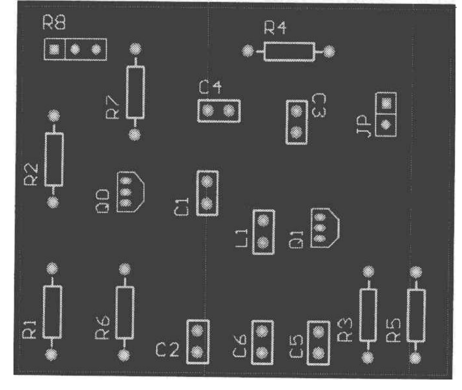


图 3-37 完成布线的电路图

4．设计规则的检测

自动布线的结果是否正确是自动布线结束后存在的一个疑问，本系统本身具备的检测功能可以来解决这个疑问。

检测的实现方法是：单击“Tools/Design Rule Check”菜单命令，弹出如图 3-38 所示对话框。

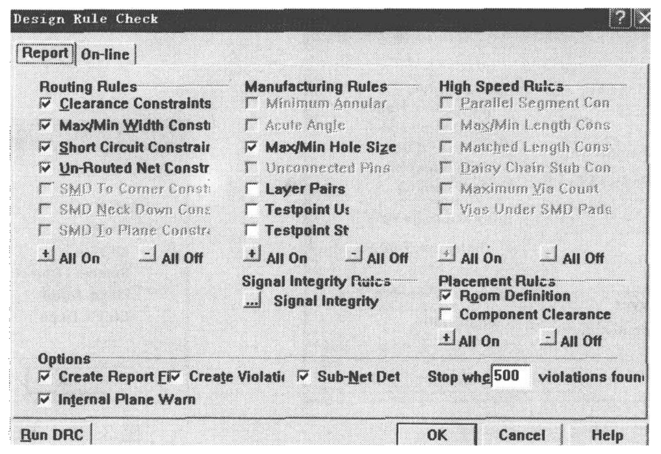


图 3-38　设计规则检测参数设置对话框

参数设置后，设计规则检测的结果有两种：一种是报表（Report）输出，可以产生检测的结果报表；另一种是在线检测（On Line）工具，也就是在布线的过程中对布线规则进行检测，防止错误产生。

5．手工调整

调整布线，增加信号输入/输出接口，加宽电源/接地线。

6．利用铺铜进行屏蔽

在高频电路中，为了增强电路抗干扰能力，通常需要大面积的铺铜区与地线相连。在放大电路的 PCB 中，如果工作在高频状态，可以通过设置铺铜来提高印制板抗干扰能力，具体操作步骤如下。

（1）单击“Place/Polygon Plane”菜单命令，放置铺铜，弹出如图 3-39 所示的铺铜设置对话框，在其中设置铺铜的参数，图中参数设置为：铺铜网络为 GND；铺铜栅格大小为 20mil；铺铜线宽为 0.2mm；连接类型为 90° 连接方式；选择圆弧形包围方式。

（2）参数设置完毕，单击“OK”按钮，工作区中出现十字光标，拖动鼠标并右击定义铺铜区的 4 个顶点，完成铺铜定义。

（3）定义完铺铜范围后，单击自动形成设置好的铺铜图形。

7．泪珠滴的使用

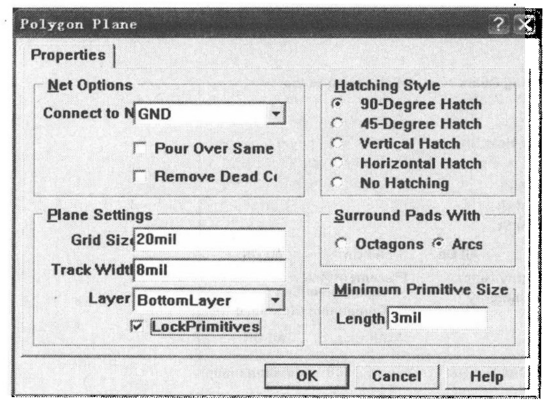
所谓泪珠滴，就是在印制导线与焊盘或过孔相连时，为了增强连接的牢固性，在连接处加大印制导线宽度。采用泪珠滴后，印制导线在接近焊盘或过孔时，线宽逐渐放大，形状就像一个泪珠，添加泪珠滴时要求焊盘要比线宽大。

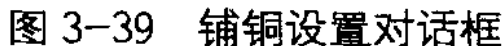
设置泪珠滴的步骤如下。

（1）选取要设置泪珠滴的焊盘或过孔，或选择网络或铜膜导线。

（2）单击“Tools→Teardrops”菜单命令，弹出泪珠滴设置对话框，如图 3-40 所示。

如图 3-40 所示，选择添加线型泪珠滴（Track），只添加选中网络的所有焊盘和过孔（All Pads，All Vias），并生成报告文件。




图 3-39　铺铜设置对话框

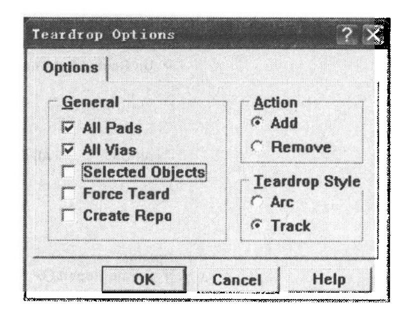


图 3-40　泪珠滴设置对话框

（3）设置完毕，单击“OK”按钮，系统自动在 VCC 网络上添加泪珠滴。

3.5.9　给电路板添加标注

1．标注文字

标注文字通常包括元件的编号、层面的作用、设计日期等。单击“Tools/Re-Annotate”菜单命令，弹出元件重新标注对话框，如图 3-41 所示。

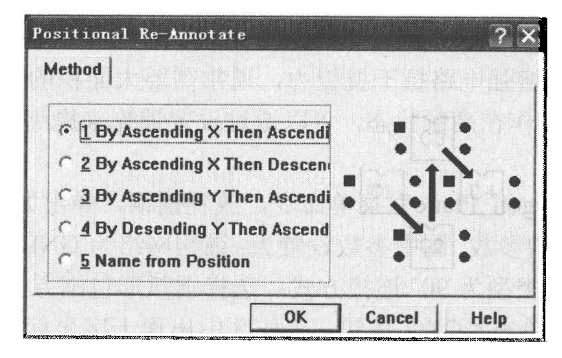


图 3-41　元件重新标注对话框

2．标注尺寸

将当前的工作层面切换成 Drill Drawing 层。单击“Place/Dimension”菜单命令以标注尺寸。

3．放置定位孔

3.5.10　PCB 图的打印输出

完成了 PCB 的设计后，就可以打印输出，从而生成 PCB 图或元件图以及其他所需的如

阻焊图、助焊图等。当使用打印机打印时，首先要对打印机进行设置，包括打印机类型、纸张大小等，然后再打印输出。

单击“File/Printer/Preview”菜单命令，系统将会生成*.prc 文件。进入该文件，单击“File/Setup Printer”菜单命令，系统将会弹出如图 3-42 所示的 PCB 打印设置对话框。设置好后单击“OK”按钮完成设置。

设置了打印机后，单击“File/Print”菜单命令，此时有 4 种选择：Print/All 打印所有图形、Print/Job 打印操作对象、Print/Page 打印由键盘所指定的页面和 Print/Current 打印当前页。根据要求选择某一项打印方案便可打印。

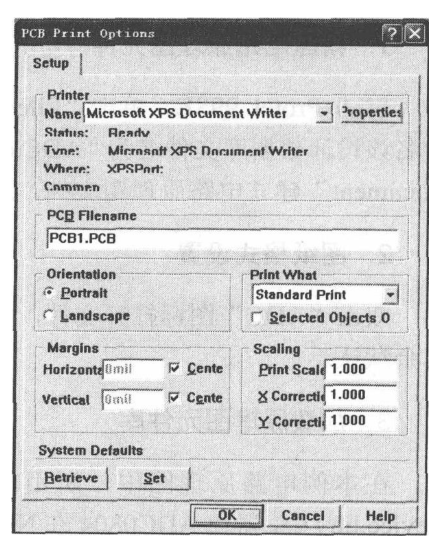


图 3-42　打印机设置对话框

3.5.11　生成元件报表

生成文件报表，用于将 PCB 上所用的元件列表。

单击“Reports/Bills/of Materials…”菜单命令，弹出元件表的向导对话框，按照自己的要求进行选择，单击“Next”按钮，最终产生元件表。

3.6　综合应用举例

下面以图 3-43 所示的“AD 变换电路”作为例子，说明如何用 Protel 99SE 绘制原理图和设计印制电路板。具体操作步骤如下。

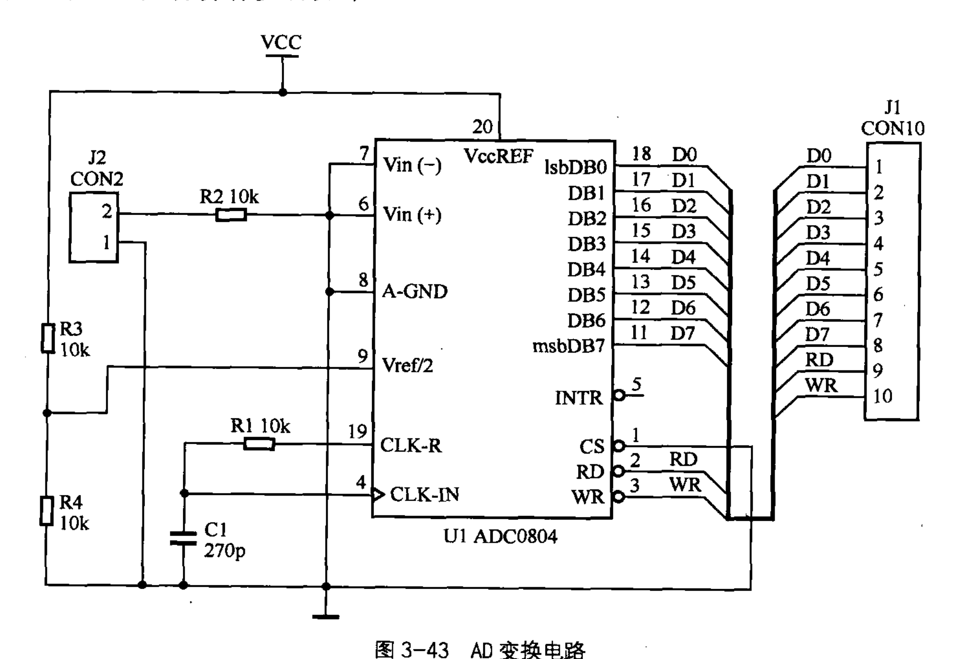


图 3-43　AD 变换电路

1．新建电路原理图元件

启动 Protel 99SE，单击“File/New…”菜单命令，新建一个设计数据库文件（*.ddb）。命名该设计数据库文件名为“ADC.ddb”。再次单击“File/New…”菜单命令，选择“Schematic Document”建立电路原理图文档，新建一个电路原理图文件，并将文件名命名为“ADC”。

2．图纸格式设置

双击“ADC”图标打开文件，单击“Design/Options”菜单命令，设置图纸大小为 A4，其余默认。

3．加载原理图元件库

在本例电路原理图中，所用的元件电阻、电容和插头在基本元件库（Miscellaneous Device.lib）中，器件 ADC0804 在 NSC databooks.ddb 元件库中。单击设计管理器中的“Browse Sch”选项，然后单击“Add/Remove”按钮，选中这两个元件库，单击“OK”按钮即可完成元件库的添加。

4．放置元件

依据前面介绍的方法，使用原理图元件库编辑器编辑原理图中元器件。如编辑电阻，在设计库管理器窗口左边的元件库面板的 Library 栏中选择“Miscellaneous Device.lib”，然后在“Components In Library”中利用滚动条找到 RES 并选定它。接下来单击“Place”按钮，此时出现一个随鼠标移动的 RES 符号，将符号移动到适当位置单击使其定位即可。

5．连接导线

所有元件放置结束后就可以开始连线了，单击“Place/Wire”菜单命令将编辑状态切换到连线模式，按照电路原理图进行连线。如图 3-44 所示是连接好的电路原理图。

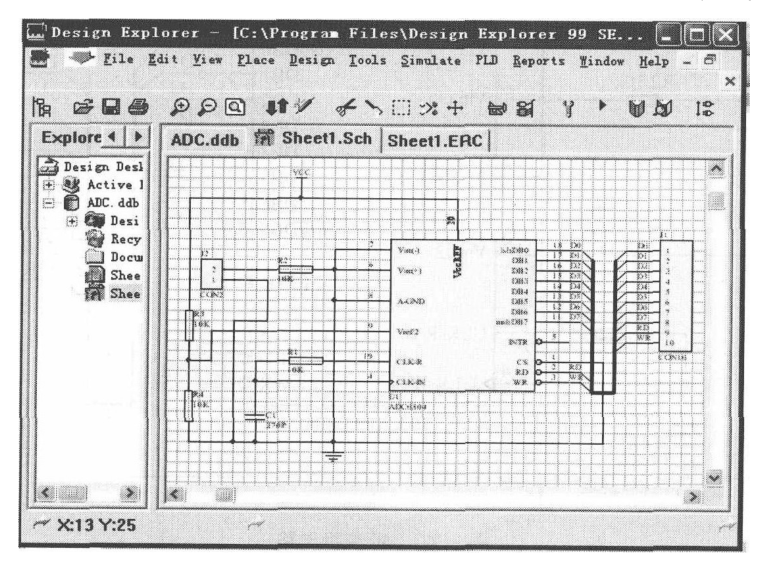


图 3-44 完成的电路原理图

6. 自动标注编号

单击“Tool/Annotate”菜单命令对元件自动编号。

7. 电气规则检查

单击“Tools/ERC”菜单命令，对画好的电路原理图进行电气规则检查，检查完毕后，显示 ERC 报告文件。如图 3-45 所示，如没有错误，就可以进入下一步。

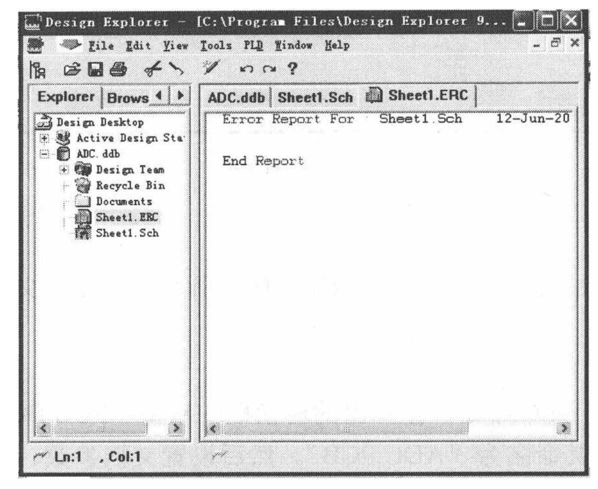
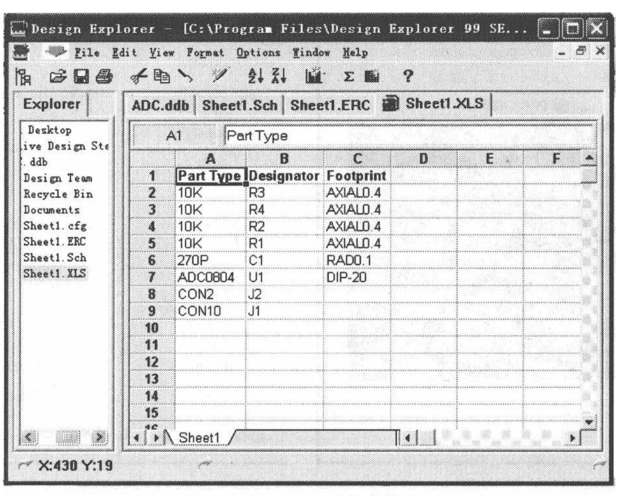


图 3-45 ERC 报告文件

8. 生成元件清单

单击“Reports/Bill of Material”菜单命令，按照提示向导进行操作就可以得到如图 3-46 所示元件清单，该清单是 Excel 格式的电子表格。



	A	B	C
1	Part Type	Designator	Footprint
2	10K	R3	AXIAL0.4
3	10K	R4	AXIAL0.4
4	10K	R2	AXIAL0.4
5	10K	R1	AXIAL0.4
6	270P	C1	RAD0.1
7	ADC0804	U1	DIP-20
8	CON2	J2	
9	CON10	J1	

图 3-46 元件清单

9. 建立网络表

单击“Design/Create Netlist”菜单命令，建立如图 3-47 所示的网络表。

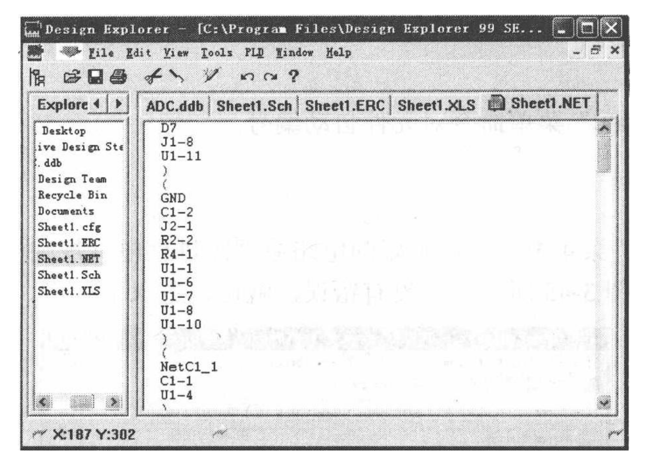


图 3-47 网络表

10. 保存文件

完成电路设计后，保存电路原理图的.Sch 文件和所产生的相关文件。

11. 新建 PCB 文件

单击“File/New…”菜单命令，在弹出的对话框中双击“PCB Document”图标，创建一个印刷电路图文件并将其命名为“ADC.PCB”。然后设置文档参数、工作系统参数和所需的元件封装库。

在工作窗中单击打开该文件，即可进入印制电路板的编辑器，图 3-48 所示为绘制好边界的电路板编辑器。

在这里要提醒一点，这是双层板，所以要注意工作层面的设置，图 3-49 所示为设置好的工作层面。

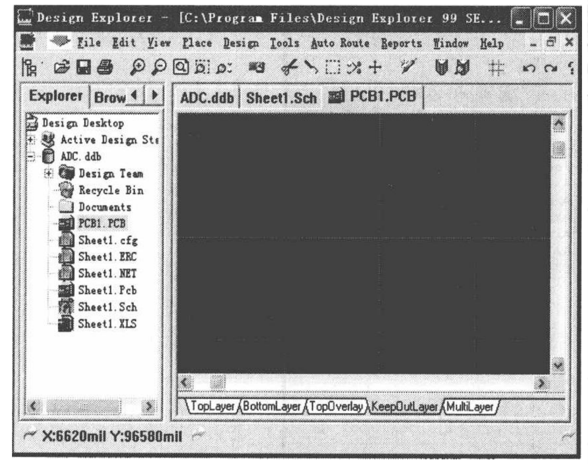


图 3-48 印制电路板的编辑器

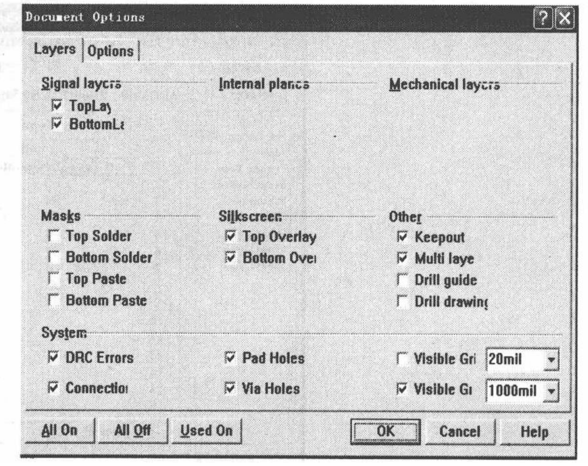


图 3-49 工作层面的设置

12. 装载元件封装库

在浏览器的组合窗中选择库“Library”，单击“Add/Remove”按钮，弹出关于引入库文

件的对话框。选择所需要的元件封装库，单击“OK”按钮即可。

13. 利用设计同步器装入网络表和元件

在原理图编辑器中单击“Design/Update PCB”菜单命令，或者利用网络表文件装入网络表和元件，即在 PCB 编辑器中单击“Design/Load Nets”菜单命令。

14. 元件自动布局与布线

由于图中元件不多，进行手动布局即可。手动布局完成后就可以进行自动布线。单击“Auto Route/All”菜单命令可对整体进行自动布线，最后需要做手工调整。布线结束后，如有需要可添加标注。完成的 PCB 图如图 3-50 所示。

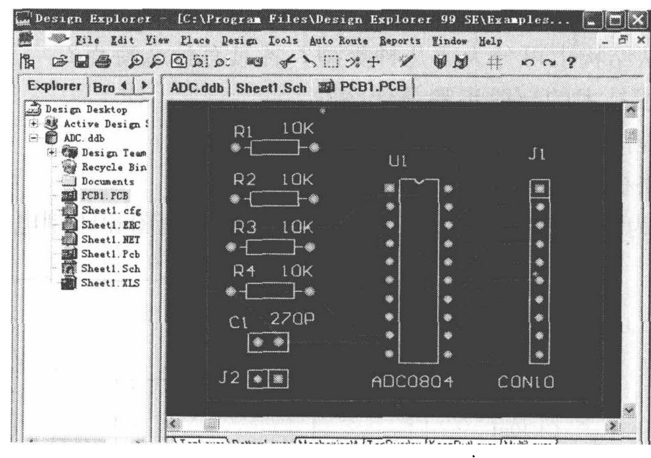


图 3-50 PCB 图

15. 打印输出

原理图和 PCB 图都设计结束后，单击“File/Printer”菜单命令就可以打印输出了。

16. 注意事项

(1) 原理图设计。

① 在设计原理图时，切记要仔细阅读各个电路元件的 datasheet 文档，在文档中需要获得以下信息：元器件的定义、功用、结构、在电路及 PCB 中的连接方式、封装、注意事项等。

② 明确器件的各个引脚的含义、在哪种情况下有效（如高电平有效）以及其连接方式。例如，单片机的中断是低电平有效，而其触发信号来自高电平有效的引脚，则两者之间需通过一个反相器来连接。

③ 放置滤波电容的情况：在电源、逻辑器件和模拟器件附近放置 104 滤波电容器用于滤波，其中模拟器件附近同时还得加上一个 10pF 的电解电容器共同滤波。

(2) PCB 设计。

① 遵循“先大后小，先难后易”的布置原则，将重要的单元电路、核心元器件优先布局。

② 布局中遵循“组件模块化”，参考原理框图，根据单板的主信号流向规律安排器件。

③ 布局中的一些要求：总的连线尽可能短而少，模拟信号与数字信号要分开，高电压、

大电流的强信号与低电压、小电流的弱信号要分开，高频信号与低频信号要分开，高频元器件的间隔要充分。

④ 相同结构的电路部分尽量采用“对称式”布局。

⑤ 晶振的附近少放器件。

⑥ 带极性的器件要注意按极性妥善放好，否则可能引起爆炸。

⑦ 复杂电路的模拟部分选择双面铺地，同时在组件密度高的周围打孔将地线连接，可减少地线阻抗。

⑧ 滤波电容要紧随需要滤波的组件。

习　题　三

1．印制电路板从结构上可分为哪几种类型？各具有什么特点？

2．PCB 自动布线技术的一般步骤是什么？

3．简述绘制原理图的基本步骤。

4．简述印制电路板的设计步骤。

5．图 3-51 所示为单级晶体管放大电路，试绘制出原理图和印制电路板。

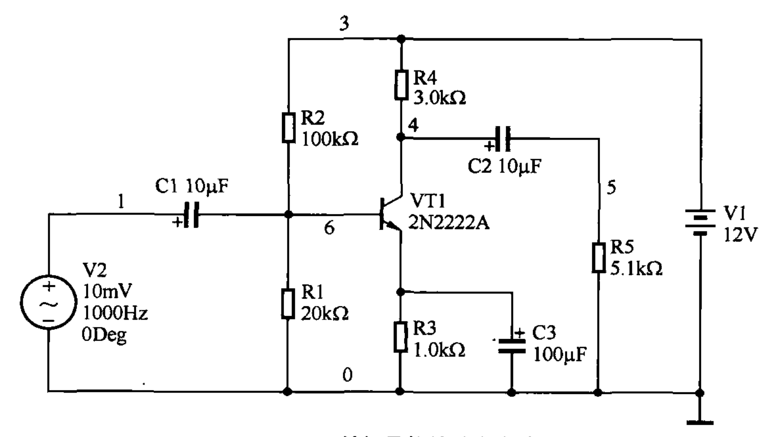


图 3-51　单级晶体管放大电路

6．图 3-52 所示为单管放大电路，试画出原理图和印制电路板。

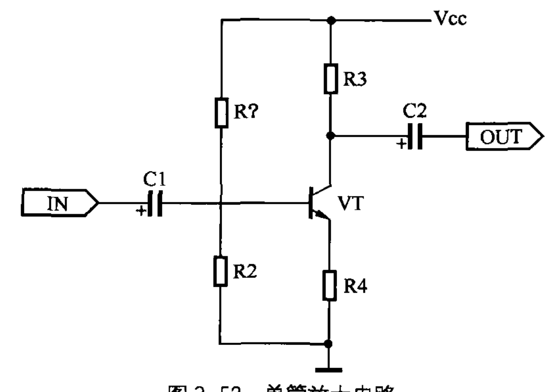


图 3-52　单管放大电路

第 4 章 可编程逻辑器件应用技术

本章要点

- 可编程逻辑器件的特点与分类
- 数字电路设计基本组件及其 VHDL 模型
- 基于 VHDL 的设计方法和步骤
- Quartus II 开发软件的应用

4.1 可编程逻辑器件概述

随着微电子设计技术和工艺的发展，数字集成电路从电子管、晶体管、中小规模集成电路、大规模及超大规模集成电路（Very Large Scale Integrated Circuit，VLSIC）逐步发展到今天的专用集成电路（Application Specific Integrated Circuit，ASIC）。ASIC 的出现降低了电子产品的生产成本，提高了系统的可靠性，缩小了电子设计的物理尺寸，但 ASIC 设计周期长，改版投资大等缺陷制约了它的应用范围。

全新的可编程逻辑器件（Programmable Logic Device，PLD）正越来越多地替代 ASIC 用于数字电路的设计和前端数字信号处理的运算。可编程逻辑器件具有 ASIC 相同的特点，例如：在规模、重量和功耗等方面都有所降低；吞吐量更高、能够更好地防止未授权复制，元器件和开发成本进一步降低，开发时间也大大缩短；具有在系统可重复编程的特性，从而可以产生更为经济的设计。

通常设计 ASIC 电路需要额外的半导体处理步骤，而可编程逻辑器件不需要这些步骤。这些额外步骤能够提供更高级别的、更高性能的 ASIC，但同时也增加了一次性工程成本（Non-reoccurring Engineering，NER）。可编程门阵列解决方案的设计者可以完全控制设计的实现过程，而不需要任何实际的集成电路制造设备或者因为后者而延缓设计进度。

可编程逻辑器件随着微电子制造工艺的发展取得了长足的进步。从早期的只能存储少量数据，完成简单逻辑功能的可编程只读存储器（Programmable ROM，PROM）、紫外线可擦除只读存储器（Ultraviolet Erasable PROM、UEROM）、电可擦除只读存储器（Electrically Erasable Programmable ROM，EEPROM）发展到能完成中规模和大规模数字逻辑功能的可

编程逻辑阵列（Programmable Logic Array，PLA）、可编程阵列逻辑（Programmable Array Logic，PAL）、通用阵列逻辑（Generic Array Logic，GAL），现在发展到复杂可编程逻辑器件（Complex Programmable Logic Device，CPLD）和现场可编程门阵列（Field Programmable Gate Array，FPGA）。随着半导体工艺的发展与市场需要，超大规模、超高速、低功耗的新型 FPGA/CPLD 不断涌现市场。新一代的 FPGA 甚至集成了中央处理器（Central Processing Unit，CPU）和数字信号处理器（Digital Signal Processer，DSP）内核，可以在一片 FPGA 上进行软硬件协同设计，为实现片上可编程系统（System on a Programmable Chip，SoPC）提供了强有力的硬件支持。

广义上讲，可编程逻辑器件是指一切通过软件手段更改、配置器件内部连接结构和逻辑单元，完成既定设计功能的数字集成电路。目前常用的可编程逻辑器件主要有简单的逻辑阵列（PAL/GAL）、复杂可编程逻辑器件（CPLD）和现场可编程逻辑阵列（FPGA）等 3 大类。

PAL/GAL 密度较低，具有的低功耗、低成本、高可靠性、软件可编程、可重复更改等特点。在很多简单的数字逻辑以及很多对成本十分敏感的设计中，GAL 等简单的可编程逻辑器件仍然被大量使用。GAL 等器件发展至今已经近 30 年了，新一代的 GAL 以功能灵活、封装小、成本低、可重复编程、应用灵活等优点仍然在数字电路领域扮演着重要的角色，越来越多的 74 系列逻辑电路被 GAL 取代。目前比较大的 GAL 器件供应商主要是 Lattice 公司。

CPLD 是在 GAL 的基础上发展起来的，其基本结构由可编程 I/O 单元、可编程逻辑宏单元、布线池和其他辅助功能模块构成。CPLD 可实现的逻辑功能比 GAL 有了大幅度的提升，一般可以完成较复杂、较高速度的逻辑功能设计，如接口转换、总线控制等。CPLD 的主要器件供应商有 A1tera、Lattice 和 Xilinx 等公司。

FPGA 的基本组成部分包括可编程 I/O 单元、基本可编程逻辑单元、丰富的布线资源、嵌入式 RAM 块、底层嵌入功能单元、内嵌专用硬核等。FPGA 的集成度很高，器件密度从数万门到数千万门不等，可以完成极其复杂的时序与组合逻辑电路功能，适用于高速、高密度的高端数字逻辑电路设计。FPGA 的主要器件供应商有 Xilinx、Altera、Lattice、Actel 和 Atmel 等公司。

可编程器件的开发可以用原理图输入的方法设计，也可以用硬件描述语言（HDL）的方法设计。HDL 的可移植性好，使用方便，但效率不如原理图；原理图输入的可控性好，效率高，比较直观，但设计大规模 CPLD/FPGA 时显得很烦琐，移植性差。

在复杂的 FPGA /CPLD 设计中，通常采用原理图和文本结合的方法来设计，适合用原理图的地方就用原理图，适合用文本的地方就用文本，并没有强制的规定。在最短的时间内，用最熟悉的工具设计出高效、稳定并符合设计要求的电路才是设计人员最终目的。

本章介绍的硬件描述语言是 VHDL。VHDL 的英文全名是 Very-High-Speed Integrated Circuit Hardware Description Language，诞生于 1982 年。1987 年底，VHDL 被 IEEE 和美国国防部确认为标准硬件描述语言。

VHDL 主要用于描述数字系统的结构、行为、功能和接口。除了含有许多具有硬件特征的语句外，VHDL 的语言形式和描述风格与句法十分类似于一般的计算机高级语言。VHDL 的程序结构特点是将一项工程设计（或称设计实体，可以是一个元件，一个电路模块或一个

系统）分成外部（或称可视部分，即端口）和内部（或称不可视部分，即涉及实体的内部功能和算法完成部分）。在对一个设计实体定义了外部界面后，一旦其内部开发完成后，其他的设计就可以直接调用这个实体。这种将设计实体分成内外两部分的概念是 VHDL 系统设计的基本点。

有 VHDL 基础的读者可以略过学习本章 4.2 节内容，直接学习本章 4.3 节数字电路设计基本组件及其 VHDL 模型。可将本章 4.2 节作为 4.3 节的查阅内容。

4.2 VHDL 要素

任何一种程序设计语言都规定了自己的一套符号和语法规则，而程序就是由这些定义的符号按照规定的语法规则写成的。在程序中使用的符号若超出规定的范围或不按语法规则书写，都被视为非法，计算机将无法识别。

4.2.1 VHDL 文字

VHDL 文字主要包括数值型文字和标识符，数值型文字主要有数字型、字符串型、位串型。

1. 数字型文字

数字型文字的值有多种表达方式，现列举如下：

（1）整数文字。由数字和下划线组成，如 7，156e2($=156\times10^2$)，45_234_287（=45 234 287）。数字间的下划线用来将数字分组，提高文字的可读性。

（2）实数文字。由数字、小数点和下划线组成，如 18.993，670_551.453(=670 551.453)。

（3）以数制的基和指数表示的文字：用这种方法表示的数由 5 个部分组成。第一部分，用十进制数标明数制进位的基数；第二部分，数制隔离符号“#”；第三部分，以基表达的整数；第四部分，指数隔离符号“#”；第五部分，用十进制表示的指数部分，这一部分的数如果是 0，可省去不写。现举例如下：

10#170#　　　--（十进制数表示，等于 170）；
2#11111110#　--（二进制数表示，等于是 254）；
16#1AF#E-2　--（十六进制数表示，等于 $431\times10^{-2}=4.31$）；
16#1AF#E2　 --（十六进制数表示，等于 $431\times10^{2}=43100$）。

（4）物理量文字：VHDL 综合器不接受此类文字。

例如：50s（50 秒），200m（200 米），177A（177 安培）。

2. 字符串文字

字符串文字包括字符和字符串。字符是用单引号引起来的 ASCII 字符，可以是数值，也可以是符号或字母，如‘R’，‘0’，‘*’，‘b’。而字符串则是一维的字符数组，须放在双引号中。VHDL 中有两种类型的字符串：文字字符串和数位字符串。

（1）文字字符串。文字字符串是用双引号引起来的一串文字，如“BB$CC”，“ERROR”，“BOTH S AND Q EQUAL TO L”，“X”都是文字字符串。

（2）数位字符串。数位字符串也称位矢量，是预定义的数据类型 bit 的一位数组。

数位字符串的表示首先是计算基数，然后将该基数表示的值放在双引号中，基数符以“B”、“O”、和“X”表示，并放在字符串的前面。它们的含义分别如下：“B”为二进制基数符号，表示二进制数位 0 或 1，在字符串中每一个位表示一个 bit。“O”为八进制基数符号，在字符串中的每一个数代表一个八进制数，即代表一个 3 位（bit）的二进制数。“X”为十六进制基数符号，代表一个十六进制数，即字符串中每一位代表一个 4 位的二进制数。

例如：

```
Data1<=B"1_1101_1111"        -- 二进制数数组，位矢数组长度是 9；
Data2<=O"58"                 -- 八进制数组，位矢数组长度是 6；
Data2<=X"AB0"                -- 十六进制数数组，位矢数组长度是 12。
```

3. 标识符

标识符用来定义常数、变量、信号、端口、子程序或参数的名字。VHDL 的基本标识符是由 26 个大小写英文字母、数字 0～9 以及下画线“_”组成的字符串。

注意：

（1）标识符以英文字母开头，不连续使用下划线“_”，不能以下画线“_”结尾；

（2）识符中的英语字母不区分大小写；

（3）VHDL 的保留字不能用于作为标识符使用。

例如：decoder_1，fft，sig_n，not_ack，state0，idle 是合法的标识符。而_decoer_1，2fft，sig_#n，not—ack，ryy_rst_，data_ _ bus，return 则是非法的标识符。

4. 段名及下标名

段名用于指示数组型变量或信号的某一段元素，而下标名则用于指示数组型变量或信号的某一元素。

段名的格式为：

```
标识符(表达式 1 to/downto  表达式 2)；
```

下标名格式为：

```
标识符(表达式)
```

表达式的数值必须在数组元素下标号范围以内，并且是可计算的。“to”表示是数组下标序列由低到高，如“3　to　8”；“downto”表示数组下标序列由高到低，如“9　downto　2”。

如果表达式是一个可计算的值，则此操作可很容易地进行综合。如果是不可计算的，则只能在特定情况下综合，且耗费硬件资源较大。

下标名及段名使用示例如下：

```
signal   a,b,c: bit_vector(0 to 5);
signal   m: integer  range 0 to 5;
signal   y,z:bit;
y <=a(m);                      --下标名（m 是不可计算型下标表示）；
```

```
z<=b(3);                    --下标名(3是可计算型下标表示);
b(0 to 3)<=a(4 to 7);       --下标段名(以段的方式进行赋值);
b(4 to 7)<=a(0 to 3);       --下标段名(以段的方式进行赋值);
```

4.2.2 VHDL 中的数据类型

VHDL 是一种强类型语言，要求设计实体中的每一个常数、信号、变量、函数及设定的各种参量都必须具有确定的数据类型。只有数据类型相同的量才能互相传递和作用。作为强类型语言可以使 VHDL 编译或综合工具很容易地找出设计中的各种常见错误。

1. VHDL 的预定义数据类型

VHDL 的预定义数据类型都是在 VHDL 标准程序包 Standard 中定义的，在实际使用中，自动包含在 VHDL 的源文件中，不必通过 use 语句调用。

（1）布尔（boolean）数据类型。

布尔数据类型是一个二值枚举型数据类型，取值有 false 和 true 两种。虽然布尔量也是二值枚举量，但它和“位”不同，没有数值的含义。

（2）位（bit）数据类型。

位数据类型也属于枚举型，取值只能是 1 或 0。位数据类型的数据对象，如变量、信号等，可以参与逻辑运算，运算结果仍是位的数据类型。

（3）位矢量（bit_vector）数据类型。

位矢量是用双引号括起来的数字序列，如“0100”。位矢量是基于 bit 数据类型的数组，用位矢量数据来表示总线状态十分方便、形象。使用位矢量时必须注明位宽，即数组中的元素个数和排列。例如：

```
signal a: bit_vector(0 to 7);
```

信号 a 被定义为一个具有 8 位位宽的矢量，它的最左位是 a（0），最右位是 a（7）。

（4）字符（character）数据类型。

字符类型通常用单引号括起来，如‘A’，‘a’，‘6’。VHDL 对标识符中的字母大小写不敏感，但字符类型是区分大小写的，如‘B’不同于‘b’。

（5）整数（integer）数据类型。

整数类型的数包括正整数、负整数和零。在 VHDL 中，整数的取值范围是−2 147 483 647～+2 147 483 647，即取值范围是$-(2^{31}-1)\sim+(2^{31}-1)$。

（6）自然数（natural）和正整数（positive）数据类型。

自然数是整数的一个子类型，非负的整数，即零和正整数；正整数是不包括 0 的整数。

（7）实数（real）数据类型。

VHDL 的实数类型类似于数学上的实数，或称浮点数。实数的取值范围为−1.0e38～+1.0e38。由于实数类型的实现相当复杂，目前在电路规模上难以承受，实数类型仅能在 VHDL 仿真器中使用，VHDL 综合器不支持实数。

（8）字符串（string）数据类型。

字符串数据类型是字符数据类型的一个约束型数组，也称为字符串数组。字符串必须用双引号标明。如：“good lucky”、“010011”。

（9）时间（time）数据类型。

时间数据类型是 VHDL 中唯一预定义的物理类型。完整的时间类型包括整数和物理量单位两部分，整数和单位之间至少留一个空格。如 50 ms，30 ns。在系统仿真时，时间数据非常有用，设计人员用它表示信号的时延，以便更好的模拟实际系统的运行环境。

（10）错误等级（severity_level）。

在 VHDL 仿真器中，错误等级用来指示设计系统的工作状态。错误等级共有 4 种可能的状态值：note（注意）、warning（警告）、error（出错）、failure（失败）。在仿真过程中，可输出这 4 种值来提示被仿真系统的当前工作情况。

2．IEEE 预定义标准逻辑位与矢量

在 ieee 库的程序包 std_logic_1164 中，定义了两个非常重要的数据类型，标准逻辑位（std_logic）和标准逻辑矢量（std_logic_vector）。在数字电路的描述中，经常用到这两种数据类型。

（1）标准逻辑位 std_logic 数据类型。

在 VHDL 中，定义了 9 种逻辑值，分别为‘u’（未初始化），‘x’（强未知的），‘0’（强 0），‘1’（强 1），‘z’（高阻态），‘w’（弱未知的），‘l’（弱 0），‘h’（弱 1），‘-’（忽略）。

在程序中使用此数据类型前，须先声明，即要加入下面的语句：

```
library  ieee;
use ieee.std_logic_1164.all;
```

在仿真和综合中，std_logic 值是非常重要的，它可以使设计者精确模拟一些未知和高阻态的线路情况。对于综合器，高阻态和‘-’忽略态可用于三态门的描述。但就综合而言，std_logic 型数据能够在数字器件中实现的只有其中 4 种值，即“-”，“0”，“1”和“z”。当然，这并不表明其余的 5 种值不存在。这 9 种值对于 VHDL 的行为仿真都有重要意义。

（2）标准逻辑矢量（std_logic_vector）数据类型。

标准逻辑矢量通常在数字逻辑电路用于描述总线。

std_logic_vector 是定义在 std_logic_1164 程序包中的标准一维数组，数组中的每一个元素的数据都是标准逻辑位 std_logic。

3．用户自定义数据类型方式

VHDL 允许用户自定义新的数据类型。用户自定义数据是用类型定义语句 type 和子类型定义语句 subtype 实现的。

（1）type 语句用法。

type 语句语法结构如下：

```
type  数据类型名  is  数据类型定义;
type  数据类型名  is  数据类型定义 of 基本数据类型;
```

其中，数据类型名由设计者自定，数据类型定义部分用来描述定义数据类型的表达方式和表

达内容，关键词 of 后的基本数据类型是指数据类型定义的元素的基本数据类型，一般是取已有的预定义数据类型，如 bit，std_logic 或 integer 等。例如：

```
type stl is array(0 to 15)of  std_logic;
type week is(sun,mon,wed,tues,thu,fri,sat);
```

第一句定义的数据 st1 是一个具有 16 个元素的数组型数据类型，数组中的每一个元素的数据类型都是 std_logic 型；第二句所定义的数据类型是由一组文字表示的，其中的每一个文字都代表具体的数值。

在 VHDL 中，任一数据对象都必须归属某一数据类型，只有同数据类型的数据对象才能进行相互作用。利用 type 语句可以完成各种形式的自定义数据类型，以供不同类型的数据对象间的相互作用和计算。

（2）subtype 语句用法。

子类型 subtype 只是由 type 所定义的原数据类型的一个子集，它满足原始数据类型的所有约束条件，原数据类型称为基本数据类型。子类型 suptype 的语句格式如下：

```
subtype  子类型名 is  基本数据类型  range  约束范围;
```

子类型定义中的基本数据类型必须是在前面已通过 type 定义的类型。例如：

```
subtype week is integer range 0 to 6;
```

其中，integer 是标准程序包中已定义过的数据类型，子类型 week 只是把 integer 约束到只含 7 个值的数据类型。

子类型的定义只在基本数据类型上作一些约束，并没有定义新数据类型。因此，属于子类型的和属于基本数据类型的数据对象间的赋值和被赋值都可以直接进行，不必进行数据类型的转换。

利用子类型定义数据对象，除了使程序提高可读性及便于处理外，其实质性的好处在于可提高综合的优化效率。这是因为综合器可以根据子类型所设的约束范围，有效地推知参与综合的寄存器的最合适的数目。

4.2.3　VHDL 数据对象

在 VHDL 中，将可以赋予一个值的对象称为数据对象。VHDL 是一种硬件描述语言，而硬件电路的工作过程实际上是信号传输并被处理至输出的过程，所以 VHDL 最基本的数据对象就是信号。为了便于描述硬件电路，还定义了另外两类数据类型：变量和常数。在电子电路设计中，这 3 类数据对象都具有一定的物理含义。信号对应地代表物理设计中某一条硬件连接线；常数对应地代表数字电路的电源和接地等；变量对应关系不太直接，通常只代表暂存某些值的载体。

1. 常数（CONSTANT）

常数的声明和设置主要是为了使程序更容易阅读和修改。例如，将位矢的宽度定义为一个常数，只要修改这个常量就能很容易地改变宽度，从而改变硬件结构。在程序中，常数是一个恒定不变的值，一旦作了数据类型的赋值定义后，在程序中不能再改变，因而具有全局

意义。常量的声明格式如下：

```
constant 常量名：数据类型：=表达式；
```

例如：

```
constant  fbus: bit_vector: ="010011";
constant  delay := 25ns;
```

注意：VHDL 要求所定义的常量数据类型必须与表达式的数据类型一致。

2．变量（VARIABLE）

在 VHDL 语法规则中，变量是一个局部量，只能在进程和子程序中使用。变量不能将信息带出对它作出定义的当前设计单元。

任何变量都只有声明后才能使用。变量在声明时，可以赋初值，也可以到变量使用时，用变量赋值语句再进行赋值。

变量声明语句的格式如下：

```
variable  变量名：数据类型：=初始值；
```

变量赋值语句的格式为：

```
目标变量名：=表达式；
```

例如：

```
variable   a: integer;          --定义 a 为整数型变量
variable   b, c: integer: = 0; --定义 b 和 c 为整型变量，初始值为 0
```

变量的赋值是一种理想化的数据传输，不存在任何延时的行为。变量在赋值时不能产生附加时延。例如 a，b，c 都是变量，则使用下面语句产生时延是错误的。

```
a:=b+c after 10ns;
```

3．信号（SIGNAL）

信号是电子电路内部硬件实体相互连接的抽象表示，是描述硬件系统的基本数据对象。它类似于连接线，可用于内连元件。事实上，信号除了没有方向说明以外，与实体的端口（port）概念是一致的。相对于端口来说，其区别只是输出端口不能读入数据，输入端口不能被赋值。信号可以看成是实体内部的端口。或者说，实体的端口只是一种隐形的信号，端口的定义实际上是作了隐式的信号定义，并附加了数据流动的方向，即端口本质上也是一种信号。

信号的定义格式如下：

```
signal 信号名：数据类型：=初始值；
```

以下是信号的定义示例：

```
signal s1: std_logic :=0;       --定义了一个标准位的单值信号 s1，初始值为低电平。
signal s2, s3: bit;             --定义了两个位 bit 的信号 s2 和 s3，未赋初值。
```

当信号定义了数据类型和表达方式后，在 VHDL 设计中就能对信号进行赋值了。信号的赋值语句表达式如下：

```
目标信号名 <=表达式;
```

这里的表达式可以是一个运算表达式，也可以是数据对象（变量、信号、常数）。代入符“<=”表示赋值操作，即将数据信息传入。同时信号可以附加延时。例如：

```
x<=5;
s1<=s2 after 10ns;              -- 源信号 s2 经过 10ns 后传给目标信号 s1
```

变量赋值与信号赋值的区别在于：变量具有局部特征，它的有效性只局限于所定义的一个进程/子程序中，是一个局部的暂时性数据对象，对于它的赋值是立即发生的（假设进程已启动）；信号具有全局性特征，它不但是一个设计实体内部各单元之间数据传递的载体，而且设计实体可通过信号与其他实体进行通信。信号的赋值并不是立即发生的，它发生在一个进程结束时，赋值过程总是有些延时，也由此反映了硬件系统的重要特性，因为综合后可以找到与信号对应的硬件结构。

常数、变量、信号 3 者的使用比较如下。

（1）从硬件电路系统来看，常数相当于电路中的恒定电平，如 GND 或 Vcc 接口。而变量和信号则相当于组合电路系统的门电路与门电路间的连接及其连线上的信号值。

（2）从行为仿真和 VHDL 语句功能上看，变量和信号的区别主要表现在接受和保持信号的方式、信息保持与传递的区域范围上。信号可以设置延时量，而变量则不能；变量只能作为局部的信息载体，而信号则可作为模块间的信息载体。变量的设置有时只是一种过渡，最后的信息传输和不同组件间的通信都靠信号来完成。

（3）从综合后所对应的硬件电路结构来看，信号一般将对应更多的硬件结构，但在许多情况下，信号和变量没有太大的区别。例如，在满足一定条件的进程中，综合后它们都能引入寄存器。这时它们都具有能够接受赋值这一重要的共性，而 VHDL 综合器并不理会它们在接受赋值时存在的延时特性。

（4）虽然 VHDL 仿真器允许变量和信号设置初始值，但在实际应用中，VHDL 综合器并不会把这些信息综合进去。这是因为实际的 FPGA/CPLD 芯片在上电后，并不能确保其初始状态的取向。因此，就时序仿真来说，设置的初始值在综合时没有实际意义。

4.2.4 VHDL 的运算操作符

VHDL 的各种表达式由操作数和操作符组成，其中操作数是各种运算的对象，而操作符则用于规定运算的方式。在 VHDL 中共有 4 类操作符，分别可以进行逻辑运算、关系运算、算术运算和并置运算。各种操作符所要求的操作数的类型详见表 4-1。

表 4-1　　　　VHDL 操作符列表

优先级别	类　　型	操作符	功　　能	操作数数据类型
	逻辑运算符	and	逻辑与	bit，boolean，std_logic
		or	逻辑或	bit，boolean，std_logic
		nand	逻辑与非	bit，boolean，std_logic
		nor	逻辑或非	bit，boolean，std_logic
低		xor	逻辑异或	bit，boolean，std_logic
		nxor	异或非	bit，boolean，std_logic
	关系运算符	=	等号	任何数据类型
		/=	不等号	任何数据类型
		<	小于	枚举与整数类型及对应的一维数组
		>	大于	枚举与整数类型及对应的一维数组
		<=	小于等于	枚举与整数类型及对应的一维数组
		>=	大于等于	枚举与整数类型及对应的一维数组
	逻辑运算符	sll	逻辑左移	bit 或布尔型一维数组
		srl	逻辑右移	bit 或布尔型一维数组
		sla	算术左移	bit 或布尔型一维数组
		sra	算术右移	bit 或布尔型一维数组
		rol	逻辑循环左移	bit 或布尔型一维数组
		ror	逻辑循环右移	bit 或布尔型一维数组
	加、减、并置运算符	+	加	整数
		−	减	整数
		&	并置	一维数组
	正负运算符	+	正	整数
		−	负	整数
	乘、除法运算符	*	乘	整数和实数（包括浮点数）
		/	除	整数和实数（包括浮点数）
高		mod	取模	整数
		rem	取余	整数
		**	指数	整数
		abs	取绝对值	整数
		not	取反	bit，boolean，std_logic

各种操作符的使用说明如下。

（1）严格遵循在基本操作符间的操作数是同数据类型的规则，严格遵循操作数的数据类型必须与操作符所要求的数据类型完全一致的规则。

（2）注意操作符之间的优先级别。当一个表达式中有两个以上的算术符时，可使用括号将这些运算分组。

（3）在使用乘法运算符时，应该特别慎重，因为它可以使逻辑门数大大增加。

（4）并置运算符“&”的操作数的数据类型是一维数组，可以利用并置运算符将普通操作数或数组组合起来形成新的数组。例如，“vh” & “dl” 的结果为 “vhdl”；“1” & “0” 的结果为 “10”。连接操作常用于字符串，但在实际运算过程中，要注意并置操作前后的数组长度应一致。

4.2.5 VHDL 的属性描述

VHDL 没有一般程序设计语言中的运算类标准函数，取而代之的是多种能反映和影响硬件行为的属性。VHDL 的属性可分为数值类、函数类、信号类、类型类和范围类等属性，这里讨论函数类和信号类的属性描述。

1. 函数类属性

函数类属性以函数的形式向设计人员提供数据类型、数组、信号的相关信息。

（1）数据类型的属性函数。

利用数组属性可以获得数组的区间，该属性的格式为：

```
object'succ(x);                --获取 x 的下一个值
object'pred(x);                --获取 x 的前一个值
object'leftof(x);              --获取 x 的左边值
object'rightof(x);             --获取 x 的右边值
```

其中，“object” 为数据类型名，x 为其中的一个元素。

例如：

```
type time is (year,month,day,hour,min,sec);
```

则

```
time'pred(hour);               --获取元素 hour 的前一个值 day
time'leftof(day);              --获取元素 day 的左边值 month
```

（2）数组的属性函数。

该属性的格式为：

```
object'left(n);                --获取索引号为 n 的区间左端边界值
object'right(n);               --获取索引号为 n 的区间右端边界值
object'high(n);                --获取索引号为 n 的区间高端边界值
object'low(n);                 --获取索引号为 n 的区间低端边界值
```

其中，“object” 为数组名；n 为多维数组中所定义的多维区间的序号。默认值 n=1，表示对一维空间进行操作。

（3）信号的属性函数。

利用信号属性可得到信号的行为和功能信息，该属性的格式为：

```
object'event;                  --反映信号的值是否变化，若是，则返回为“真”
object'active;                 --反映信号是否活跃，若是，则返回为“真”
object'last_event;             --反映从最近一次事件到现在所经过的时间，返回一个时间值
object'last_value;             --反映信号变化前的取值，并将该值返回
object'last_active;            --反映从最近一次活跃到现在所经过的时间，返回一个时间值
```

信号的事件（event）和活跃（active）是两个不同的概念，必须严格区分。信号的活跃定义为信号值的任何变化。信号值由 1 变为 0 是一个活跃，而从 1 变为 1 也是一个活跃，唯一的准则是发生了事情。信号的事件则要求信号值发生变化。信号值从 1 变为 0 是一个事件，但从 1 变为 1 虽是一个活跃却不是一个事件。所有的事件都是活跃，但并非所有的活跃都是事件。

2．信号类属性

信号类属性的作用对象是信号，其返回值也是一个信号。共有 4 种信号类属性，分别是 delayed (time)，stable (time)，quiet(time)和 transaction。

delayed (time)：即延时，该属性使受它作用的信号延时 time 所规定的值。

如 a'delayed (5 ns)表示信号 a 延时 5 ns。

stable (time)：用于监测信号在规定时间内的稳定性，若受它作用的信号在 time 所规定的时间内没有发生事件，则该属性的结果为“true”。

Quiet(time)：用于监测信号在规定时间内的是否“安静”，若受它作用的信号在 time 所规定的时间内没有发生事情或事件（active 或 event），则该属性的结果为“true”。

transaction：用于检测信号的 active 或 event，当 active 或 event 发生时，该属性的值也将发生改变。

4.3 数字电路设计基本组件及其 VHDL 模型

译码器、多路选择器、串并/并串转换电路、锁存器/触发器、计数器、有限状态机等是典型的数字电路设计组件，本节以这些电路的 VHDL 模型与设计为例，引出相关的 VHDL 结构、语法表述、数据规则和语法特点，并加以详细说明。

4.3.1 多路选择器和译码器的 VHDL 模型及相关语法

在 CPLD/FPGA 设计中，译码器和多路选择器可以用来管理大量的数据和控制信号，是一个非常有用的功能。

1．多路选择器的 VHDL 描述

多路选择器是一种简单的组合逻辑，可对一系列控制信号进行译码，并由此产生选择信号，选择输入信号中的一个作为输出。如一个 n 比特地址选择线可以对 2^n 路信号进行选择。2 选 1 数据选择器输入 1 位地址，对 2（即 2^1）路信号进行选择输出。其符号如图 4-1 所示，功能如表 4-2，例如，图 4-1 所示是其 VHDL 完整描述。

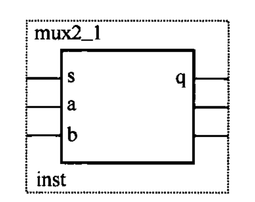


图 4-1 二选一复用器实体图

表 4-2 二选一数据选择器功能表

s（地址选择端）	q（输出端）
0	a
1	b

例如，表 4-1 所示为二选一多路复用器 VHDL 程序。

```
library ieee;                              --库声明
use ieee.std_logic_1164.all;
entity mux2_1 is                           --实体定义
  port(s : in std_logic;                   --端口声明以及端口的定义
      a : in std_logic;                    --端口名:端口模式 数据类型
      b : in std_logic;
      q : out std_logic);                  --注意最后一个端口说明语句不要加分号!
end;
architecture behav of mux2_1 is            --结构体定义,
 begin
      q<=a when  (s='0')  else             --结构体描述
         b when  (s='1')  else             -- =为数据比较符号，比较结果为布尔量。
         'X';                              --std_logic逻辑中，'X' 定义为强未知
end;
```

程序说明：VHDL 代码中的“--”表示注释。凡是和“--”在同一行，并且在“--”后面的内容都是注释内容。VHDL 不区分大小写（连保留字也不区分大小写）。

由图 4-1 可见，VHDL 描述由 3 大部分组成，分别为库声明、实体和结构体。

（1）库声明：库是专门存放预先编译好的程序包（package），这样它们就可以在其他设计中被调用。VHDL 程序设计中常用的库有 ieee 库、std 库和 work 库，其中 VHDL 程序需使用 ieee 定义的库中的内容时，需要在使用前加上库声明语句，而使用 std 库和 work 库中定义的内容时，无需提前声明。

例如在上述二选一多路复用器设计中开始部分有：

```
library ieee;
use ieee.std_logic_1164.all;
```

ieee 是国际电工电子委员会的认定的标准库的标志名，下面一个 use 语句使得设计单元可使用程序包 std_logic_1164 中预定义的内容。

ieee 库是 VHDL 设计中最常用的资源库，包含 ieee 标准的 std_logic_1164、numeric_bit、numeric_std 以及其他一些支持工业标准的程序包。其中最常用的是 std_logic_1164 程序包，大部分程序都是以此程序包中设定的标准为设计基础。此外，还有一些程序包虽非 ieee 标准，但由于它们已成为事实上的工业标准，也并入了 ieee 库。这些程序包中，常用的有 std_logic_arith、std_logic_signed、std_logic_unsigned 程序包。

std 库是 VHDL 的标准库，定义了 VHDL 类型，如 bit（位）、bit_vector（位矢量）等。VHDL 在编译过程中会自动调用这个库，所以使用该库时不需要用库声明语句声明。

work 库是用户在进行 VHDL 设计时的现行工作库，即为设计该工程文件所保存的文件夹目录。用户的设计成果将自动保存在这个库中，是用户自己的仓库。用户设计项目的成品、半成品模块以及先期已设计好的元件都放在其中。同 std 库一样，使用该库不需要预先说明。

std 库和 ieee 库中所包含的程序包见表 4-3。

表 4-3　　Std 库和 ieee 库中所包含的程序包说明

库名	程序包名	包中预定义内容
std	standard	是 VHDL 的标准库，为 VHDL 语言中的类型和函数所做的预定义。在编译过程中会自动调用这个库，所以使用时不需要用语句另外说明
ieee	std_logic_1164	ieee 库中最常用的程序包，是 ieee 的标准程序包。定义了 VHDL 语言描述互联数据类型的标准。常用的数据类型是 std_logic（标准逻辑位）、std_logic_vector（标准逻辑位矢量）
ieee	numeric_std	定义了一组基于 std_logic_1164 中定义的类型算术运算符，如"+"、"−"等
ieee	std_logic_arith	预先编译在 ieee 库中。主要是在 std_logic_1164 程序包的基础上扩展了 unsigned（无符号）、signed（符号）数据类型、转换函数和用于 unsigned 与 signed 类型的算术/比较操作
ieee	std_logic_signed	预先编译在 ieee 库中，主要定义有符号数的运算，重载后可用于 integer（整数）、std_logic 和 std_logic_vector 之间的混合运算，并且定义了 std_logic_vector 到 integer 的转换函数
ieee	std_logic_unsigned	允许 std_logic_vector 类型使用 unsigned 类型一样的操作

库声明中，如果使用 use ieee.std_logic_unsigned.all，则矢量被看作无符号整数；如果使用 use ieee.std_logic_signed.all，则矢量被看作带符号整数。对于前者，有（"1001" > "0000"）成立；对于后者，有（"1001" < "0000"）成立。

（2）实体（entity）：实体是设计中最基本的模块。它描述了系统与外部结构进行信息交换的端口和参数，是系统外部结构的描述。

端口名赋予实体（或系统）引脚的名称，一般用几个英文字母组成，或者英文字母加数字。每个端口信号名在实体中是唯一的。端口模式定义了引脚的输入输出方向，可综合的（即能将 VHDL 程序综合成可实现电路）端口模式有 in（输入到实体）、out（从实体输出）、inout（双向，此端口既可输入也可输出）和 buffer（缓冲）4 种。buffer 端口可以反馈到实体内部。

2 选 1 数据选择器的实体对应的原理图符号如图 4-1 所示。s、a、b 为输入端口，数据类型为标准逻辑；q 为输出端口，数据类型为标准逻辑 std_logic。

在此例中端口数据类型取自 IEEE 标准库，因为使用了该库定义的数据类型，所以在实体说明前增加了库说明语句。

常用的数据类型有 integer 类型（整型）、boolean 类型（布尔型）、std_logic 类型（标准逻辑类型）和 bit 类型（位类型）等。关于数据类型的介绍详见第 4.2.2 小节。

小技巧：采用 VHDL 进行设计时，建议信号变量端口尽量使用 std_logic 类型或其派生类型。这样做是为了统一信号格式，信号连接方便，不容易出错误，尤其是模块与模块之间的连接。因为一旦采用其他类型，例如 bit，则与此相连接的所有线都得定义成 bit，人为增加麻烦。

（3）结构体（architecture）：结构体用于描述实体的内部逻辑功能和电路结构。

结构体具体指明了该设计实体的行为，定义了设计实体的功能。由例 4-1 可以看出，该电路的逻辑功能描述是用 when-else 结构的并行语句表达的。

条件判断语句 when-else 通过测定表达式（s= '0'）的比较结果，以确定由哪一路端口向 q 赋值。

when_else 结构语句的用法如下：

当某一条件满足时，对一个信号赋值，则可以使用代码完成单个赋值。

```
output <= value when condition;   --条件成立(为真)，则赋值；否则比较下一条件。
```

用 else 语句将上面的语句扩展，可以覆盖更多的不同条件，结构如下：

```
output <= value1 when condition1 else
          value2 when condition2 else
          …
          valuen when conditionn;
```

最后，如果有一个“覆盖所有”的条件，那么增加的最后一条语句如下：

```
output <= value1 when condition1 else
          value2 when condition2 else
          …
          valuen when condition n else
          valuedefault;
```

当条件（condition）是表达式时表达式须用括号“()”括起来，使代码更为清晰。除了 when-else 结构，也可以用其他的语句形式来描述多路选择器的逻辑行为。

小技巧：when-else 语句具有优先级，第一个 when 条件级别最高，最后一个最低。可用 when-else 语句实现有优先级的逻辑电路。

（4）组合逻辑电路的多种 VHDL 建模方法。

描述组合逻辑有多种描述方式，下面还以简单的组合逻辑 2 选 1 多路选择器为例，对组合逻辑的直接逻辑运算描述、when-else 语句描述、if-then-else 语句描述和 case 语句描述加以比较说明。

【例 4-1】 2 选 1 多路选择器直接逻辑运算描述。

```
library ieee;
use ieee.std_logic_1164.all;
entity mux2_1 is
    port(s : in std_logic;
        a : in std_logic;
        b : in std_logic;
        q : out std_logic);
end;
architecture rtl of mux2_1 is
begin
    q <= (not s and a ) or ( s and b );    --not、and、or 为逻辑操作符非、与、或
end;
```

程序说明：

① 此例中，and、or、not 是逻辑操作符。VHDL 有 and（与）、or（或）、nand（与非）、nor（或非）、xor（异或）、xnor（同或）、和 not（非）7 种基本逻辑操作符。逻辑操作符对应的操作数（操作对象）的数据类型有 bit、boolean（布尔类型，取值为真或假）和 std_logic。

② 绝大多数情况下结构体由 process(进程)构成。一条信号赋值实际上就是一个 process，其敏感信号为右边所有信号。相当于：

```
process(s,a,b)      --(s,a,b)为敏感信号列表
begin
   q <=(not s and a)or(s and b);
end process;
```

process 语句是一种并发处理语句，在一个构造体中多个 process 语句可以同时并发运行，符合硬件电路的特征，因此，process 语句是 VHDL 中描述硬件系统并发行为最常用、最基本的语句。进程语句的一般格式为：

```
进程名:  process  (敏感变量)
   进程说明语句:
begin
   顺序描述语句;
end process  进程名;
```

敏感变量是用来启动进程的，process 显示敏感列表必须完整。敏感变量发生变化（如由‘0’变‘1’或由‘1’变‘0’），将启动该 process 语句。一旦进程启动后，process 中的语句将从上至下逐句执行一遍。当最后一个执行完毕以后，返回到开始的 process 语句，等待下一次启动。因此，只要 process 中指定的敏感变量变化一次，该 process 语句就会执行一遍。

若构造体中有多个进程存在，各个进程之间的关系是并行关系；进程之间的通信则通过接口由敏感变量传递，并行地同步执行。process 内部各语句之间是顺序关系。在系统仿真时，process 语句是按书写顺序一条一条向下执行的。

【例 4-2】 2 选 1 多路选择器 if-then-else 语句描述。

```
library ieee;
use ieee.std_logic_1164.all;
entity mux2_1ifelse is
  port(s : in std_logic;
       a : in std_logic;
       b : in std_logic;
       q : out std_logic);
end;
architecture behav of mux2_1ifelse is
begin
      process(s,a,b)              -- (s,a,b)为敏感信号列表
      begin
      if s = '0' then
         q <=  a ;                --端口 a 的数据向端口 q 传输
      else  q <=  b;
      end if;
      end process;
end;
```

小技巧：if 必须有一个 else 对应，当没有 else 语句时将产生不希望的锁存器，即不完全条件语句将综合成锁存器。具体说明见第 4.3.2 小节：锁存器/触发器/寄存器的 VHDL 模型。

【例 4-3】 2 选 1 多路选择器 case 语句描述。

```
library ieee;
```

```
use ieee.std_logic_1164.all;
entity mux2_1case is
  port(s : in std_logic;
       a : in std_logic;
       b : in std_logic;
       q : out std_logic);
end;
architecture behav of mux2_1case is
begin
      process(s,a,b)            -- (s,a,b)为敏感信号列表
      begin
      case s is
          when '0' => q <= a;
          when '1' => q <= b;
          when others => q <='X';
      end case;
      end process;
end;
```

case 语句的功能是从众多不同语句序列中选择其中之一执行。

case 语句的结构如下：

```
case  条件表达式  is
    when  条件表达式的值 => 顺序处理语句;
    when  条件表达式的值 => 顺序处理语句;
    …
end case;
```

当执行到 case 语句时，首先计算条件表达式的值，然后根据条件语句中与之相同的选择值，执行对应的顺序语句，最后结束 case 语句。条件语句中的“=>”相当于“then”的作用。

以上 4 种 VHDL 建模方法的对比：

直接逻辑运算描述又称为寄存器传输级描述，是与硬件一一对应的方法，要求设计者对具体硬件电路比较熟悉。when-else 语句描述、if-then-else 语句描述和 case 语句描述可称为行为级建模描述，该方法利用条件分支，比较符合人的常规思维，有利于加快设计速度及减轻维护负担。一般情况下推荐使用行为级描述风格。

小技巧：

① case 语句中所有表达式的值都是并行处理的；case 语句所有表达式的值都必须穷举且不能重复，不能穷尽的值用 others 表示；

② 在 if 语句中，先处理最起始的条件，如果不满足，再处理下一个条件；在 case 语句中，没有值的顺序号，所有值是并行处理的，在 when 项中的值只能出现一次，且不能重复使用。与 if 语句相比，case 语句组的程序可读性比较好。但对相同的逻辑功能，综合后，case 语句比 if 语句的描述耗用更多的硬件资源。有的逻辑，case 语句无法描述，只能用 if 语句来描述。这是因为 if-then-elsif 语句具有条件相与的功能和自动将逻辑值“-”（忽略）包括进去的功能（逻辑值“-”有利于逻辑化简），而 case 语句只有条件“或运算”的功能。

③ If -elsif -elsif -else 的优先级中最后一个 else 优先级最低，在高频设计中，必须把关键路径放在优先级高的语句中，也就是靠前的 if-elsif 中。

2．译码器

译码器可以将某一种形式的数字表示转换成另一种形式的数字表示，如将一个 n 比特的输入译码为 2^n 个独立的逻辑信号。2-4 译码器输入 2 个逻辑信号，将它转换为 $4(2^2)$ 个输出信号中的一个，即 4 个输出信号中只有一个表示为当前被选择的输入。其符号如图 4-2 所示，功能如表 4-4，例 4-4 是其 VHDL 建模描述。

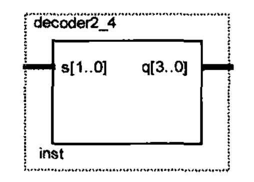


图 4-2　2-4 译码器实体图

表 4-4　　2-4 译码器功能表

s1	s2	q3	q2	q1	q0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

【例 4-4】 2-4 译码器 VHDL 程序。

```
library ieee;                                        --库声明
use ieee.std_logic_1164.all;
entity decoder2_4 is                                 --实体定义
  port(s : in std_logic_vector(1 downto 0);          --端口声明以及端口的定义
       q : out std_logic_vector(3 downto 0));        --端口名：端口模式 数据类型
end;                                                 --注意最后一个端口说明语句不要加分号！
architecture behav of decoder2_4 is                  --结构体定义
begin
       q<="0001" when  (s="00")  else                --结构体描述
          "0010" when  (s="01")  else                -- “=” 为数据比较符号
          "0100" when  (s="10")  else
          "1000" when  (s="11")  else
          "XXXX";
end;
```

程序说明：

数据类型 std_logic_vector 为标准逻辑矢量，是一个一维数组，数组中的每一个元素的数据都是以上定义的标准逻辑位 std_logic。逻辑矢量是用双引号括起来的数字序列。

前面以 2 选 1 多路选择器为例进行了结构体的不同描述，2-4 译码器可以仿照 2 选 1 多路选择器做出相应的多种描述。

4.3.2　锁存器/触发器/寄存器的 VHDL 模型及相关语法

锁存器/触发器是同步时序数字系统的基本组成单元，也是寄存器传输级（Register Transfer Level，RTL）设计的基础。

1．锁存器

锁存器可以简单的定义成一种电平敏感器件，也就是说，其输出仅仅依赖于输入的值。最常用的是 D 锁存器，输出端 q 仅在使能端 en 为高电平时才随输入端 d 变化（也可以低电

平使能），其符号如图 4-3 所示，功能表如表 4-5 所示，例 4-5 是其 VHDL 完整描述。

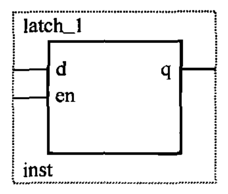


图 4-3 锁存器实体图

表 4-5 锁存器功能表

en	q
0	保持
1	d

【例 4-5】 锁存器的 VHDL 描述。

```
library ieee;
use ieee.std_logic_1164.all;
entity latch_1 is
  port(d  : in std_logic;
       en : in std_logic;
       q   : out std_logic);
end;
architecture behav of latch_1 is
begin
  process(d,en)
    begin
    if (en = '1') then              --en 为 1，送数、锁存
       q <=d;
    end if;                         --缺省 else，保持
  end process;
end;
```

输出 q 完全依赖于输入 d 的电平和使能信号 en，当 en 信号为高电平时，d 输出给 q。此电路为高电平触发锁存器，也称为高电平敏感锁存器。

例 4-5 描述用了一个不完整 if 语句（只有 if(en = '1')then，而没有 else），信号 d 和 en 都在敏感列表中，是个组合逻辑，但是由于 en 条件语句不完整，结果出现了一个隐含的锁存器，也就是存储电路。

利用这种不完整的条件语句的描述引进锁存器元件，从而构成时序电路的方式是 VHDL 描述时序电路的重要途径。通常，完整的条件语句只能构成组合逻辑电路，如例 4-1 中，if_then_else 语句指明了 s 为'1'和'0'全部可能的条件下的赋值操作，从而产生了多路选择器组合电路模块。

小技巧：在设计电路中，使用锁存器（latch）必须有所记录。不希望使用锁存器时应该对将条件赋值语句写全。不完整的 if 和 case 语句导致不必要的锁存器的产生。虽然在构成时序电路方面，不完整的条件语句具有独特的功能，但在利用条件语句进行纯组合电路设计时，如果没有充分考虑电路中所有可能出现的问题，即没有列全所有的条件，将导致不完整的条

件语句的描述，从而产生设计者不希望的组合与时序电路的混合体。

不完整 case 语句也会出现这样的情况，举一个简单的 VHDL 例子如下：

```
case s is
   when "00"  => y <= a;
   when "11"  => y <= b;
   when others => null;
end case;
```

在这个例子中，case 语句是不完整的，所以综合后生成的不是组合逻辑电路，而是锁存器电路（在 s 等于“01”或“10”时，y 值保持不变）。

2. 触发器

和电平触发的锁存器不同，触发器（flip-flop，FF）仅仅在使能信号或时钟信号的变化有效触发沿才可能发生状态变换。其符号如图 4-4 所示，功能如表 4-6，例 4-6 是其 VHDL 完整描述。

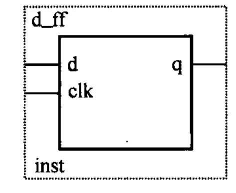


图 4-4 D 触发器实体图

表 4-6　　D 触发器功能表

clk	q
↑	D
其他	保持

【例 4-6】 D 触发器的 VHDL 描述。

```
library ieee;
use ieee.std_logic_1164.all;
entity d_ff is
  port(d   : in std_logic;
       clk : in std_logic;
       q   : out std_logic);
end;
architecture behav of d_ff is
begin
  process(clk)
    begin
    if clk'event and clk = '1' then     --检测 clk 信号的上升沿
       q <=d;
    end if;
  end process;
end;
```

输出 q 在时钟 clk 上升沿获得输入 d 的值。仿真波形如图 4-5 所示。

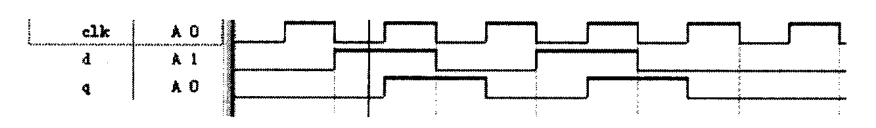


图 4-5 D 触发器仿真波形图

程序说明：程序中用到 clk'event and clk = '1'，表示 clk 信号发生变化并且变为 1，也就是说，clk 信号来了上升沿。类似的上升触发沿检测语句还有 rising_edge(clk)。若下降触发沿检测可用 clk'event and clk = '0'，或 falling_edge（clk）。

小技巧：在此程序中，d 没有在敏感列表中。触发器只有在时钟有效触发沿（这里是上升沿）到来时才会有所动作，此处可简化设计书写。当然把 d 放在敏感列表里也是正确的。

还可以将基本 D 触发器模型扩展为带异步置位和异步复位的 D 触发器。所谓异步置位和异步复位，指不管有没有时钟有效触发沿都会置位或复位。因此，置位和复位信号需要加到模型的进程敏感列表中。VHDL 代码如例 4-7 所示。

【例 4-7】 带异步置位和异步复位的 D 触发器的 VHDL 描述。

```
library ieee;
use ieee.std_logic_1164.all;
entity dff_rs is
  port(  d  : in std_logic;
       clk : in std_logic;
       rst_n: in std_logic;
       set_n: in std_logic;
       q   : out std_logic);
end;
architecture behav of dff_rs is
begin
  process(clk, rst_n,set_n)
    begin
    if (rst_n ='0') then
        q <='0';
    elsif (set_n ='0') then
        q <='1';
    elsif rising_edge(clk) then     --检测 clk 的上升沿
        q <=d;
    end if;
  end process;
end;
```

程序说明：由于 process 内部语句是顺序执行的，所以复位信号总是在置位信号之前被检查，尽管从功能上来说允许同时置位和复位，但实际上复位具有更高的优先权。

若考虑同步复位和置位，应该在时钟有效沿后立即检查置位和复位信号，程序可改为

```
process(clk, rst_n, set_n)
  begin
  if rising_edge(clk) then          --检测 clk 的上升沿
      if (rst_n ='0') then
           q <='0';
      elsif (set_n ='0') then
           q <='1';
      else   q <=d;
      end if;
  end if;
end process;
```

信号的命名要用有意义而有效的名字，如程序中信号 rst_n 和 set_n 可易看出是低电平有效的复位和置位信号。

小技巧：标识符（Identifiers）命名习惯：

① 标识符定义命名规定：标识符第一个字符必须是字母，最后一个字符不能是下画线，不许出现连续两个下划线。基本标识符只能由字母数字和下划线组成。标识符两词之间须用下划线连接，如 rst_n，set_n，Packet_addr，Data_in，Mem_wr，Mem_ce。

② 标识符大小写规定：对常量数据类型实体名和结构体名采用全部大写，对变量采用小写。对信号采用第一个词首字符大写，保留字一律小写。

③ 信号名连贯缩写的规定：长的名字对书写和记忆会带来不便，甚至带来错误。采用缩写时，应注意同一信号在模块中的一致性。一致性的缩写习惯有利于文件的阅读、理解和交流。部分缩写的统一规定为 Clk，clock；Clr，clear；En，enable；Rst，reset；Cnt，counter；Addr，address；Rd，reader；Wr，write；Reg，register；Lch，latch；Inc，increase；Mem，memory；Pst，preset。自定义的缩写最好在文件头注释。

④ 建议用有意义而有效的名字，能简单包含该信号的全部或部分信息。如输入输出信息，Data_in 总线数据输入，Din 单根数据线输入，FIFO_out 数据总线输出。如宽度信息 Cnt8_q（8 位计数器输出信号）。

⑤ 建议添加有意义的后缀，使信号名更加明确。常用的后缀如表 4-7 所示。

表 4-7　　信号命名常用后缀及其含义

_clk	时 钟 信 号
_n	求反的信号或低电平有效的信号
_en	使能控制信号
_s	模块内的反馈信号在实体端口信号名之后加后缀_s（即_same）
_L	采用 D 触发器对信号进行延迟，延迟信号的命名规则是在原信号名之后加后缀_L，_L 后加数字表示信号延迟时钟周期数，如_L1（即_lock） 若是在流水线设计中有级延迟，分别加后缀_L1 _L2…
_q	寄存器的数据输出信号
_d	寄存器的数据输入信号（即_data）
_z	连到三态输出的信号

3. 寄存器

装载和存储总线数据的一组触发器称为寄存器。寄存器和触发器的区别：寄存器具有数据输入端、时钟，通常还有复位端、装载信号（load），用来指示寄存器复位和输入数据是否装载到寄存器内部。例 4-8 设计了一个异步复位、同步送数的 8bit 寄存器。

【例 4-8】 异步复位、同步送数的 8bit 寄存器的 VHDL 描述。

```
library ieee;
use ieee.std_logic_1164.all;
entity reg is
  generic (n : natural := 7);
  port(d   : in std_logic_vector(n-1 downto 0);
```

```
        clk : in std_logic;
        rst_n: in std_logic;
        load: in std_logic;
        q   : out std_logic_vector(n-1 downto 0));
end;
architecture behav of reg is
begin
  process(clk, rst_n)
   begin
   if (rst_n ='0') then
      q <= (others=>'0');              --将q置0，也可以用q<= "00000000"
   elsif rising_edge(clk) then         --检测clk上升沿
      if (load ='1') then              --同步检测load是否等于1,若等于1,则为真,执行q<=d
         q <=d;
      end if;
   end if;
  end process;
end;
```

4.3.3 串并/并串转换电路的VHDL模型及相关语法

在现代CPLD/FPGA设计接口中，串并/并串转换电路是一个非常普遍的功能，大多数通信数据为串行方式，而大多数处理器则要求数据以并行方式存储和处理。

1. 串并转换

串并转换（Serial to Parallel Conversion，SIPO）是相当简单的处理，在时钟驱动下，将单比特的位流输入寄存器中，并依次逐步移位，直到寄存器满了为止，然后直接读取并行数据即可。例4-9是其VHDL完整描述。

【例4-9】 串并转换的VHDL描述。

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity sipo is
  generic (n : positive := 8); --类属说明，positive表示n的数据类型为正整数
  port(clk  : in std_logic;
       rst_n : in std_logic;
       din   : in std_logic;
       q     : out std_logic_vector((n-1) downto 0));
end;
architecture behav of sipo is
  signal int_reg : std_logic_vector((n-1) downto 0);    --定义int_reg的数据对象为
信号
  signal index  : integer :=0;
  signal flag   : bit;
begin
  out_process: process
  begin
  wait until rising_edge(clk); --检测clk的上升沿
```

```
    if rst_n = '0' then
          int_reg <= "00000000";  -- 或者 int_reg <= (others=>'0')
          index <=0;
    else
          int_reg(index)<= din;
          if index =7 then         --如果移进 7 位数据，则 flag 置 1，通过 flag=1 控制串并行送数
             index<=0;
             flag<='1';
          else
             index<=index+1;
          flag<='0';
          end if;
    end if;
    if flag='1' then              --flag=1，并行送数
          q<=int_reg;
    else null;
    end if;
    end process;
  end;
```

程序说明：

类属参数声明是实体说明中的可选项，其一般书写格式为

```
generic  (常数名：数据类型：=设定值;
          ...
          常数名：数据类型：=设定值);
```

类属说明：类属参数声明必须放在端口说明语句之前，是一种端口界面常数，用来规定端口的大小、实体中子元件的数目和实体的定时特性等。它和常数不同，常数只能从设计实体的内部得到赋值且不能改变，而类属参量的值可有设计实体的外部提供。因此，设计者可以从外面通过类属参数的重新设定而改变一个设计实体或一个元件的内部电路结构和规模，如指定矢量位数、器件延迟时间参数等。

例如：

```
generic (m: time := 5ns);
```

声明 m 是一个值为 5ns 的时间参数。这样，语句 tem1<= d0 and sel after m;

表示 d0 和 sel 经过"与"运算后经 5ns 延迟后才送到 tem1。

2. 并串转换

并串转换（Parallel to Serial Conversion，PISO）操作包括两个步骤，第一步是载入并行数据，第二步是寄存器内的数据在时钟的作用下逐步移出。例 4-10 是其 VHDL 建模。在这个模型中，load_n 为同步信号，低电平有效。

【例 4-10】 并串转换的 VHDL 描述。

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
```

```
entity piso is
  generic (n : positive := 8);
  port(clk  : in std_logic;
      load_n : in std_logic;
      q    : in std_logic_vector((n-1) downto 0);
      dout   : out std_logic);
end;
architecture behav of piso is
  signal int_reg : std_logic_vector((n-1) downto 0);
  signal index   : integer :=0;
begin
  out_process: process
  begin
  wait until rising_edge(clk);          --检测 clk 上升沿
  if load_n = '0' then                  --load_n 为低电平，则接收并行数据
       int_reg <= q;
       index <=0;
  else
       dout<= int_reg(index);           --通过计数器 index 实现串行逐位输出
       if index =7 then
          index<=0;
       else
          index<=index+1;
       end if;
  end if;
  end process;
 end;
```

程序说明：此程序使用时要注意 load_n 的时序控制。

4.3.4 计数器的 VHDL 模型及相关语法

计数器是实现分频器的基础，常用的计数器有普通计数器、扭环计数器。

普通计数器是加法/减法计数器，如例 4-11 所示计数器为具有异步清零的模 10 计数器，可将程序稍做修改实现其他模值得计数器。

【例 4-11】 具有异步清零的模 10 计数器 VHDL 描述。

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity counter10 is
    Port ( clk : in std_logic;
         reset _n: in std_logic;
         dout : out std_logic_vector(3 downto 0);
         c:out std_logic);
end counter10;
architecture Behavioral of counter10 is
   signal count : std_logic_vector(3 downto 0);
begin
    dout <= count;
    process(clk,reset_n)
```

```
    begin
      if reset_n='0'then                      --reset_n 实现异步清零，低电平有效
        count <= (others=>'0') ;
        c<='0';
      elsif rising_edge(clk) then             --检测 clk 上升沿
         if count = "1001" then               --计数到“1001”即 9，则清零
            count <= "0000";
            c<='1';
         else                                 --否则，加 1，继续计数
            count <= count+1;
            c<='0';
         end if;
     end if;
    end process;
end Behavioral;
```

在同一时刻，加法计数器的输出可能有多位同时发生变化，当组合逻辑对输出进行译码时，会导致竞争冒险的出现。使用扭环计数器可以避免这个问题。

扭环计数器是一种移位计数器，又称为约翰逊计数器，在每个时钟下输出只有一位发生变化。具体实现是把输出的最高位取反，然后反馈到最低位触发器的输入端。例 4-12 是具有自启动的扭环计数器的 VHDL 描述。

【例 4-12】 具有自启动的扭环计数器的 VHDL 描述。

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity johnson_counter is
    generic(n: integer :=4);
    Port ( clk : in std_logic;
         reset_n : in std_logic;
         dout : out std_logic_vector((n-1) downto 0));
end johnson_counter;
architecture Behavioral of johnson_counter is
   signal count : std_logic_vector(3 downto 0);
begin
    dout <= count;
    process(clk,reset_n)
    begin
       if (reset_n='0') then
          count <= (others => '0');
       elsif rising_edge(clk) then
          case count is
             when "0010" => count <= (others => '0');    --无效状态返回有效状态
             when "0100" => count <= (others => '0');
             when "0101" => count <= (others => '0');
             when "0110" => count <= (others => '0');
             when "1001" => count <= (others => '0');
             when "1010" => count <= (others => '0');
             when "1011" => count <= (others => '0');
             when "1101" => count <= (others => '0');
             when others => count <= not count(0) & count((n-1) downto 1);
```

```
                        --有效状态之间的转换
        end case;
      end if;
    end process;
  end Behavioral;
```

程序中的自启动是通过将无效状态转换到“0000”的有效状态中，上面程序中罗列了所有的 8 个无效状态，有些啰嗦。如果认真分析扭环计数器的有效循环和无效循环，只需要将 8 个无效状态的任何一个列出即可实现具有自启动的扭环计数器。如：

```
case count is
   when "1101" => count <= (others => '0');
   when others => count <= not count(0) & count((n-1) downto 1);
end case;
```

也可实现自启动的扭环计数器，请读者思考原因。

4.3.5 有限状态机的 VHDL 描述及相关语法

在数字电路中，有限状态机（Finite State Machines，FSM）是一种设计控制算法的基本技术，是数字电路设计的核心。有限状态机的基本概念是：存储一系列不同的状态，根据输入和当前状态在这些状态之间进行转换。状态机由状态寄存器和组合逻辑电路构成，能够根据控制信号按照预先设定的状态进行状态转移，是协调相关信号动作，完成特定操作的控制中心。有限状态机有两种类型，分别是 Moore 型（状态机的输出完全由状态变量决定）和 Mealy 型（状态机输出既与当前状态变量有关，又与输入有关）。有限状态机的一般结构如图 4-6 所示。

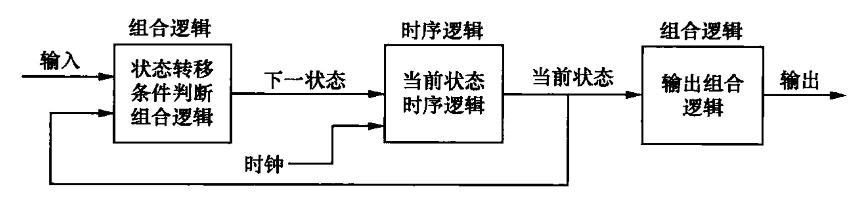


图 4-6 有限状态机的结构

从设计的观点看，描述有限状态机的方法是使用状态转移图，它可以表示状态、输出和转移条件。下面以 mealy 型状态机为例介绍状态机的设计。

【例 4-13】 mealy 型状态机的状态转移图如图 4-7 所示，请用 VHDL 建模设计该状态机。

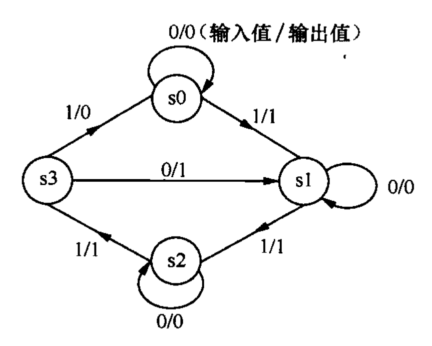


图 4-7 状态转移图

解：由状态转移图可得状态转移表，如表 4-8 所示。

表 4-8　　　　　　　　　　　　　　状态转移表

现　　态	次　　态		输　　出	
	X=0	X=1	X=0	X=1
S0	S0	S1	0	1
S1	S1	S2	0	1
S2	S2	S3	0	1
S3	S1	S0	1	0

VHDL 设计程序如下：

```
library  ieee;
use ieee.std_logic_1164.all;
entity  s_machine is
   port(clk,reset: in std_logic;
        comb_input: in std_logic;
        comb_output: out std_logic);
end entity s_machine;
architecture  behav of s_machine is
   type  states is (st0,st1,st2,st3);      --定义 states 为枚举型数据类型
   signal current_state,next_state: states;
begin
  reg: process(reset, clk)                  --时序逻辑进程
   begin
      if reset='1' then
         current_state <= st0;              --异步复位
      elsif clk= '1' and clk'EVENT then
         current_state <= next_state;       --当检测到时钟上升沿时转换至下一状态
      end if;
   end process  reg;       --由信号 current_state 将当前状态值带出此进程，进入进程 com
  com: process(current_state, comb_input)   --组合逻辑进程
   begin
      case  current_state  is              --确定当前状态的状态值
         when st0=>
            if  comb_input = '0' then       --根据外部的状态控制输入'0'
              next_state <= st0;            --在下一时钟后，进程 reg 状态维持为 st0
            else
              next_state <= st1;            --否则,在下一时钟后,进程 reg 的状态将为 st1
            end if;
            if comb_input = '0' then        --输出
               comb_output <= '0';
            else
               comb_output <= '1';
            end if;
         when st1 =>
            if comb_input = '0' then        --根据外部的状态控制输入'0'
              next_state <= st1;            --在下一时钟后，进程 reg 的状态将维持为 st1
            else
              next_state <= st2;            --否则,在下一时钟后,进程 reg 的状态将为 st2
```

```
            end if;
            if comb_input = '0' then       --输出
               comb_output <= '0';
            else
               comb_output <= '1';
            end if;
         when st2 =>
            if comb_input ='0' then
               next_state <= st2;
            else
               next_state <= st3;
            end if;
            if comb_input = '0' then       --输出
               comb_output <= '0';
            else
               comb_output <= '1';
            end if;
         when st3 =>
            if  comb_input ='0' then
               next_state <= st1;
            else
               next_state <= st0;          --否则，在下一时钟，进程 reg 的状态返回 st0
            end if;
            if comb_input = '0' then       --输出
               comb_output <= '1';
            else
               comb_output <= '0';
            end if;
       when others=>
             next_state <=s0;
     end case;
   end process com;              --由信号 next_state 将下一状态值带出此进程，进入进程 reg
end architecture  behav;
```

单击“Tools/Netlist Viewers/RTL Viewer”菜单命令，可观察到该状态机综合电路图，如图 4-8 所示。单击“Tools/Netlist Viewers/State Machine”菜单命令，可观察到该状态机综合后状态转移图，如图 4-9 所示。

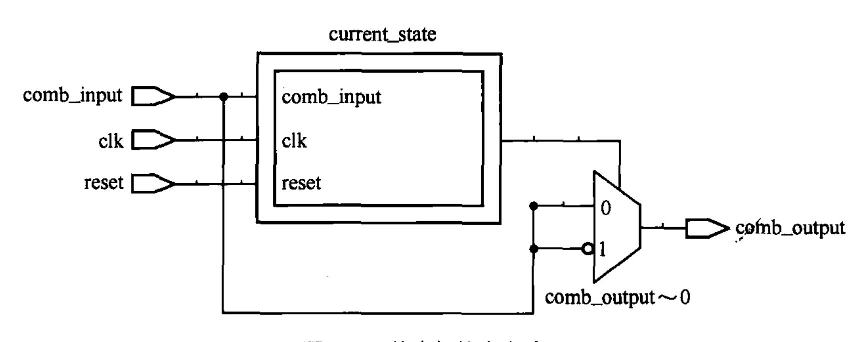


图 4-8　状态机综合电路图

设计状态机时在结构上通常遵循以下几点：

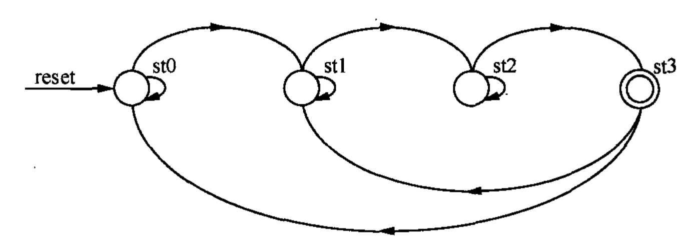


图 4-9 综合后状态转移图

（1）各进程只描述一个状态机；

（2）将无关逻辑减至最少；

（3）将状态寄存器从其他逻辑中分离出来。一般的模型由两个进程组成，一个进程用来实现时序逻辑电路，另一个进程用来实现组合逻辑电路，如果需要的话，可以使用更多的进程。

4.4 CPLD/FPGA 的设计流程

一般来说，完整的 CPLD/FPGA 的设计流程包括电路设计与输入、功能仿真、编译（逻辑综合）、综合后仿真、实现、实现后仿真与验证、板级仿真验证与调测等主要步骤。在设计一个可编程逻辑数字系统时，设计流程所包含的设计步骤如图 4-10 所示。每一个具体设计的步骤都会略有变化，但在本质上是相同的。

1. 写出一份设计规范

一个设计规范可以让每一个设计工程师去了解整个设计项目的情况以及在此项目所担当的那部分任务。一个设计规范应该包括如下信息。

（1）外部框图。外部框图包括 CPLD/FPGA 器件如何安装在系统中，这样的框图将有助于描述器件的全部功能，并且对于系统的设计者、印制电路板（PCB）设计者、系统中其他芯片的设计者和软件开发者都将是良好的参考基准。

（2）内部框图及功能模块划分。内部框图对于器件的行为描述来说是一个起点，当描述行为的 VHDL 源代码变化时，这些变化必须加到内部框图中。当内部框图中的其他因素必须更改时，这些更改也会更容易的加入描述行为的 VHDL 源代码中。

功能模块划分的主要目的是让设计层次分明、条理清晰。另外在确定功能模块划分过程中能使设计者在总体上考虑芯片的各个问题，发现一些比较深层次的问题，这对设计能否实现是非常重要的。

（3）确定关键电路时序和模块间接口时序。设计电路，尤其是数字电路，最关键的一环是设计各模块间的接口时序，确定关键电路的时序。这个工作必须在具体电路设计之前确定下来。凡是设计过电路的人都经历过，在系统联调时经常发现彼此之间的配合出了问题，当设计不能满足时序要求时，就会挖空心思地想办法改进电路，甚至更改整个设计。

时序是事先设计出来的，而不是事后测出来的。因此，我们在做总体方案时，应该深入到模块间的时序划分。关键电路的时序确定，并以此指导各项目组、项目组各成员进行设计。这项工作的实质就是确定各模块设计需求。在设计之前若选择的设计策略出现问题，则会极大增加设计实现的难度且影响产品开发的进程。

（4）定时估计。定时估计可以确定采用哪一种技术、哪一个生产厂商和哪一种器件。应该完全明白设计所需要的时钟频率以及I/O端口所要求建立时间和保持时间。

（5）设计所需资源的估计和器件封装形式的选择。若具有设计可编程器件的经验时，估计设计所需资源数目比较容易。如果没有设计经验，可以与准备要订购其器件的生产厂商进行交流，他们能帮助建立起合理的门数估计的概念，可以据此确定用哪个厂商的器件适合于设计项目。同时要了解不同的生产厂商所提供的各种可选购的封装形式，以便于选择合适的封装形式进行PCB布线设计。

（6）功耗目标。要了解影响器件功耗的因素以及器件运行时如何影响整个电路板乃至整个系统的功率消耗。确定系统需要多大的功率（包括典型情况下消耗的功率和最坏情况下所消耗的功率以及系统中每一个芯片所需要的功率），由此确定系统供电电源模块的设计。

（7）价格目标。价格目标能帮助设计者在选择芯片引脚数、芯片资源、速度、封装类型等确定必要的折中。

（8）测试程序。规划测试应该处于设计流程的初始阶段，也就是一开始就要建立起系统的易测性。否则，可能出现当设计完成，进入到测试时，却发现不能完全地或不能准确地测试器件。

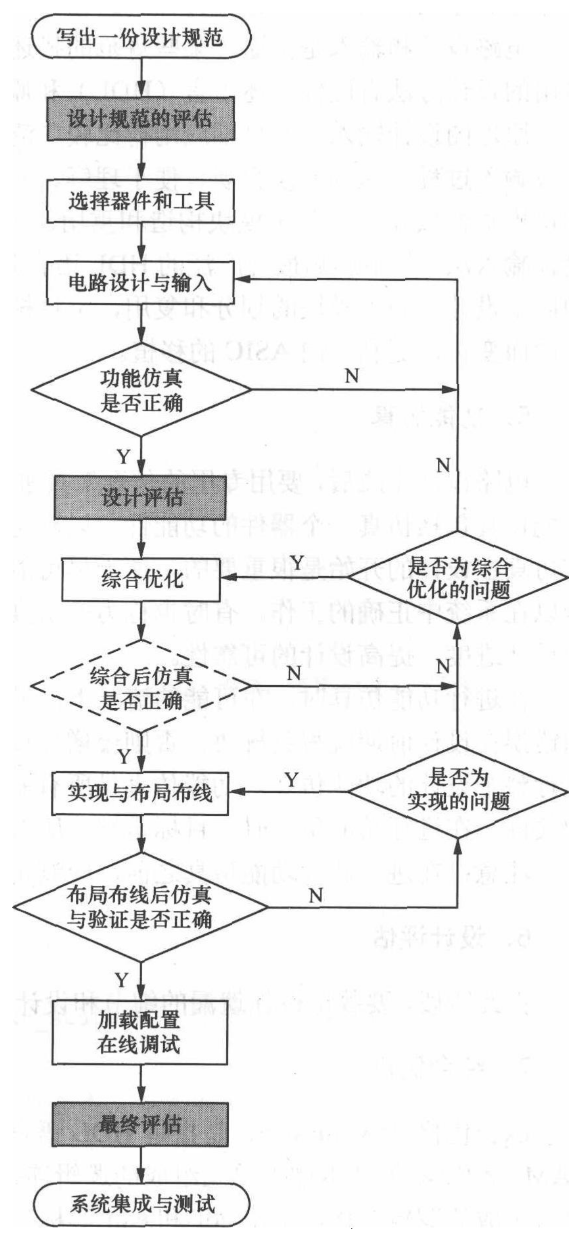


图4-10 基于FPGA/CPLD的数字系统设计流程图

2. 设计规范的评估

在制定设计规范阶段的最后，进行一次设计评估是非常重要的。因为设计规范是整个芯片的基础，与此设计项目相关人员都应该参与这一评估，找出规范中是否有错误或遗漏。

3. 选择器件和工具

有了设计规范，设计项目组就可以利用它找到最佳的可编程芯片生产商及其提供的技术，还可以根据设计，找到性价比高的器件。根据设计器件选择能够很好在一起互相配合的开发工具。

4. 电路设计与输入

在进行模块设计时，应先画出每个模块的原理结构，而后画出其工作原理时序图。在工

作原理时序图的指导下，进行具体电路设计。

电路设计和输入是指通过某些规范的描述方式，将工程师的电路构思输入给 EDA 工具。常用的设计方法有硬件描述语言（HDL）和原理图设计输入方法等。

原理图设计输入法在早期应用得比较广泛，它根据设计需求，选用器件、绘制原理图、完成输入过程。这种方法直观、便于理解、元器件库资源丰富。但在大型设计中，这种方法的可维护性较差，不利于模块构造和重用。在大型工程设计时，最常用的设计方法是 HDL 设计输入法，其中影响最为广泛的 HDL 语言是 VHDL 和 Verilog HDL。他们的特点是利用由顶向下设计，利于模块的划分和复用，可移植性好，通用性好，设计不因芯片的工艺和结构不同而变化，更利于向 ASIC 的移植。

5．功能仿真

电路设计完成后，要用专用的仿真工具进行功能仿真，验证电路功能是否符合设计需求。功能仿真包括仿真一个器件的功能性，以确定该器件是按照设计规范所描述的方式工作的。该仿真在设计的开始是很重要的，便于尽可能多的找出器件中所存在的缺陷，并且确信设计可以在系统中正确的工作，有时也称为前仿真。通过功能仿真能及时发现设计中的错误，加快设计进度，提高设计的可靠性。

在进行功能仿真时，在可能的情况下，最好是对每一个子模块进行仿真，确保 90%以上的错误在设计前期就得到解决，否则会增加设计后期的困难。当设计代码全部完成之后可以进行整个设计的功能仿真。功能仿真是所有验证环节中最重要的一环，有时是决定设计成败的关键，在进行功能仿真时，目标是解决所有的功能方面的问题。

注意：在进行芯片功能仿真之前，应制定完善的仿真测试方案，尽可能覆盖所有情况。

6．设计评估

在此阶段，要找出设计遗漏的细节和设计规范中不恰当的设想，这是最重要的评估之一。

7．综合优化

综合优化（Synthesize）是指将 HDL 语言、原理图等设计输入翻译成由和、或、非门，RAM，触发器等基本逻辑单元组成的逻辑连接（网表），并根据目标和需求（约束条件）优化所生成的逻辑连接，输出.edf 和.edn 等标准格式的网表文件，供 FPGA/CPLD 厂家的布局布线器进行实现。

8．综合仿真

综合完成后需要检查综合结果是否和设计一致，即综合后仿真。

在仿真时，把综合生成的标准延时文件反标识到综合仿真模型中去，可估计门延时带来的影响。综合后仿真虽然比功能仿真精确一些，不过只能估计门延时，不能估计线延时，仿真结果和布线后的实际情况更有一定的差距，并不十分准确。这种仿真的主要目的在于检查综合器的综合结果是否和设计输入一致。

目前主流综合工具日益成熟，对于一般性的设计，如果设计者确信自己标注明确，没有综合歧义发生，则可省略该步骤。不过如果在布局布线后仿真时发现有电路结构和设计意图

不符的现象，则常常需要回溯到综合后仿真以确认是否是由于综合歧义造成的问题。

9. 实现与布局布线

综合结果的本质是一些由和、或、非门，触发器，RAM 等基本逻辑单元组成的逻辑网表，他和芯片的实际设置情况更有较大的差距。此时应该使用 FPGA/CPLD 厂商提供的软件工具，根据所选芯片的型号将综合输出的网表适配到具体 FPGA/CPLD 器件上，这个过程就叫做实现过程。因为只有器件的开发商最了解器件的内部结构，所以实现步骤必须选用器件研发商提供的工具。在实现过程中最主要的过程是布局布线（PAR）。

所谓布局（Place），就是指将逻辑网表中的硬件原语或底层单元合理地适配到 FPGA 内部的固有硬件结构上，布局的优劣对设计的最终结果（在速度和面积两个方面）影响非常大。所谓布线（Route），是指根据布局的拓扑结构，利用 FPGA 内部的各种连线资源，合理连接各个元件的过程。布局和布线由开发软件进行操作。

一般情况下，用户能通过设置参数指定布局布线的优化准则，总的来说优化目标主要有两个方面，面积和速度。根据设计的主要矛盾，选择面积或速度或是两者平衡等优化目标，不过当两者冲突时，一般满足时序约束更重要一些，此时选择速度或时序优化目标更佳。

FPGA 的结构相对复杂，为了获得更好的实现结果，特别是确保能够满足设计的时序条件，一般采用时序驱动进行布局布线，所以对于不同的设计输入，特别是不同的时序约束，获得的布局布线结果一般有较大的差异。CPLD 结构相对简单得多，其资源有限而且布线资源一般为交叉连接矩阵，故 CPLD 的布局布线过程相对简单的多，一般称为适配过程。

10. 布局布线后仿真与检验

将布局布线的延时信息反标注到设计网表中，所进行的仿真就叫时序仿真或布局布线后仿真，也叫后仿真。该仿真的仿真延时文件包含的延时信息最全，不仅包含了门延时，还包含了实际布线延时，所以布局布线后仿真最准确，能够较好的反映芯片的实际工作情况。一般来说，布线后仿真步骤必须进行，通过布局布线后仿真能检查设计时序和 FPGA 实际运行情况是否一致，确保设计的可靠性和稳定性。但时序仿真耗时较长，同步时序逻辑设计中，时序验证主要由静态时序分析来保证。

静态时序分析是一种更快、更彻底的分析方法，它着眼于某个同步设计并确定其最高工作频率，该频率不违反任何建立和保持时间。静态定时分析软件考虑了在设计电路中从每一个触发器到其他各个触发器之间的路径以及这些路径所连接的组合逻辑，软件工具计算出最好和最坏的情况下通过这些路径所需要的延时。任何违反一个触发器所需的建立和保持定时要求的路径，或者其延时超过了给定时钟频率的时钟周期的路径，都将被标出来。这样，可以通过调整这些被标出的路径，使它们满足设计的时序要求。

不同阶段的仿真小结：

（1）功能仿真主要目的在于验证语言设计的电路结构和功能是否和设计意图相符。

（2）综合后仿真主要目的在于验证综合后电路结构是否和设计意图相符，是否存在歧义综合结果。

（3）布局布线后仿真主要目的是验证是否存在时序违规。

（4）时序仿真既有动态时序分析功能，又有功能验证的功能。由于时序仿真带有延时信

息，因此软件在该仿真时其运算量比功能仿真时要多得多，而且若设计改动较多时，每次功能验证都通过时序仿真来完成的话，极为费时，严重影响设计进度。因此设计的功能验证应主要由功能仿真来保证，而在同步时序电路设计中，有静态时序分析来分析一个设计是否能满足它的定时要求。

11．最终评估

如果设计组已经按照所有其他的步骤完成后，而且其他的评估也已经通过的话，最终评估只是一个最终完成的信号，表明本设计已经进行了源程序编写、仿真、逻辑综合、布局布线等操作，且本设计已经准备好进入电子系统。

12．系统集成与测试

设计研发的最后步骤就是在线调试或将生成的设置文件写入芯片中进行测试。示波器和逻辑分析仪是逻辑设计的主要调试工具。传统的逻辑功能板级验证手段是用逻辑分析仪分析信号进行的，设计时需要 FPGA 和 PCB 设计人员保留一定数量 FPGA 管脚作为测试管脚，编写 FPGA 代码时将需要观测的信号作为模块的输出信号，在综合实现时再把这些输出信号锁定到测试管脚上，然后连接逻辑分析仪的探头到这些测试管脚，设定触发条件，进行观测。

在测试中设计者还可以将一种高效的硬件测试手段（SignalTap II）和传统的系统测试方法相结合来完成，这就是嵌入式逻辑分析仪。它可以随设计文件一并存储于目标芯片中，用以捕捉目标芯片内部信号节点处的信息，而又不影响原硬件系统的正常工作，这就是 Quartus II 中 SignalTap II 的目的。在实际监测中，SignalTap II 将测得的样本信号暂存于目标器件中的嵌入式 RAM（如 ESB、M4K）中，然后通过器件的 JTAG 端口将采得的信息传出，送入计算机进行显示和分析。嵌入式逻辑分析仪 SignalTap II 允许对设计中的所有层次模块的信号节点进行测试，可以使用多时钟驱动，而且还能通过设置以确定前后触发捕捉信号信息的比例。

所有仿真或验证步骤出现问题，就需要根据错误定位返回到相应的步骤并更改或重新设计。

以上介绍的是基于 FPGA/CPLD 的数字系统设计的一般开发步骤，在具体项目中可做适当删减。一般 FPGA/CPLD 的设计步骤包括：设计目标分析与功能模块划分、确定关键电路时序和模块间接口、时序电路设计、设计验证等步骤。

4.5 用 Quartus II 完成 CPLD/FPGA 设计的实例

4.5.1 原理图、文本输入设计方法

QuartusII 软件包是 MAX+plusII 的升级版本，提供了一个完整高效的设计环境，非常适应具体的设计需要，能够支持逻辑门数在百万门以上的逻辑器件的开发，并且为第三方工具提供了无缝接口。

QuartusII 支持多种编辑输入法，包括图形编辑输入法，VHDL、Verilog HDL 和 AHDL 的文本编辑输入法，原理图编辑输入法以及状态转移图输入法。本节以 QuartusII8.0 为例，介绍基于原理图和文本的设计方法。

双击桌面上 QuartusII8.0 的快捷方式图标，启动 QuartusII8.0 软件。打开 QuartusII 集成

环境，呈现出图 4-11 所示主窗口界面。使用 QuartusII 设计电路系统之前，需要先建立设计项目（Project）。下面以模 12 计数器的设计为例，详细介绍原理图输入和文本输入的设计方法。设计步骤如下：

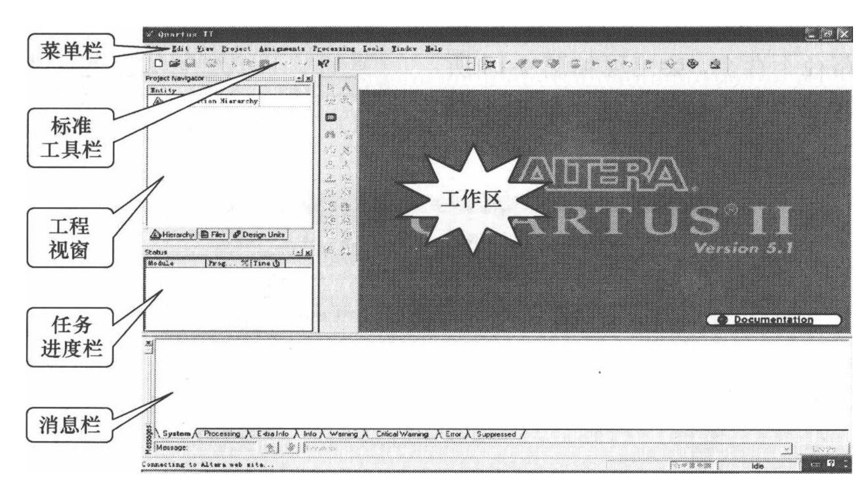


图 4-11　Quartus II 主窗口界面

1. 创建工程

在 QuartusII 集成开发环境下，执行菜单命令“File/New Preject Wizard”，出现如图 4-12 所示的对话框，该对话框显示用向导设计新建工程所需的步骤。

（1）建立工程文件。单击“Next”按钮，出现“工程设置”对话框，如图 4-13 所示。在此对话框中设置工程路径、工程名和顶层实体名。

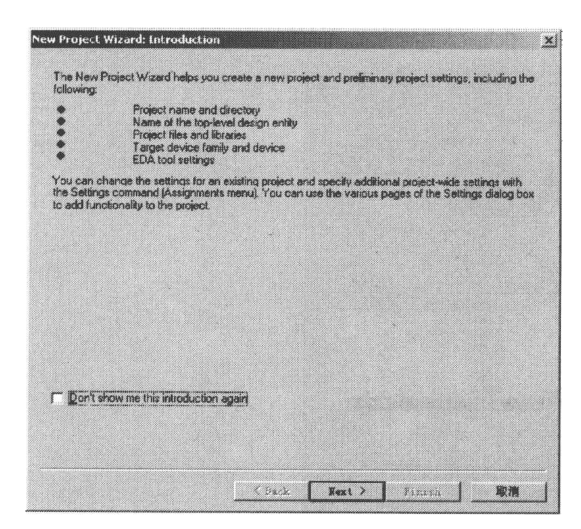


图 4-12　新建工程向导介绍

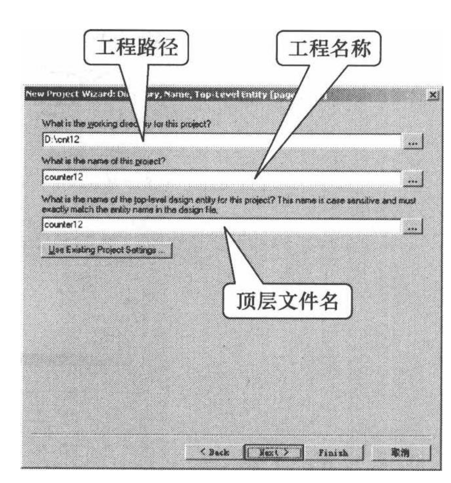


图 4-13　工程设置

第 1 个选择框为工程路径，输入工程路径为“d：\cnt12”，当然也可以选择其他目录。在第 2 个选择框输入工程名称，工程名可以取任何其他的名字。本设计只有一个文件，在此直接用顶层文件的实体名作为工程名，输入“counter12”，表示此项工程的工程名为“counter12”。

第三个选择框是当前工程顶层文件的实体名，这里输入为“counter12”，该名称和设计文件的顶层实体的名称要严格一致（Quartus II 的编译是从顶层实体开始逐级编译的，也就是说顶层实体是整个编译过程的入口）。单击“Next”按钮，如果指定路径不存在，会出现如图 4-14 所示的消息框，单击“是”按钮，进入如图 4-15 所示对话框。

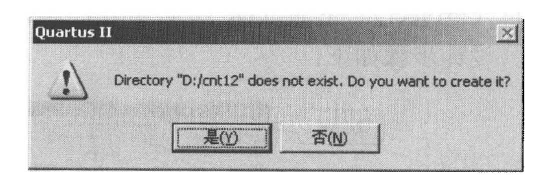


图 4-14　为工程新建目录

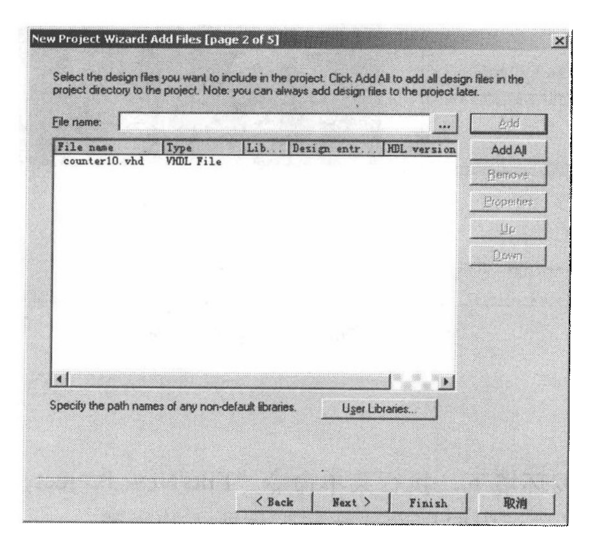


图 4-15　为工程添加设计文件

（2）加入工程文件。在这一步，向导要求向新项目加入已存在的设计文件。因为设计文件还没有建立，所以单击“Next”按钮，跳过这一步。弹出如图 4-16 所示对话框。

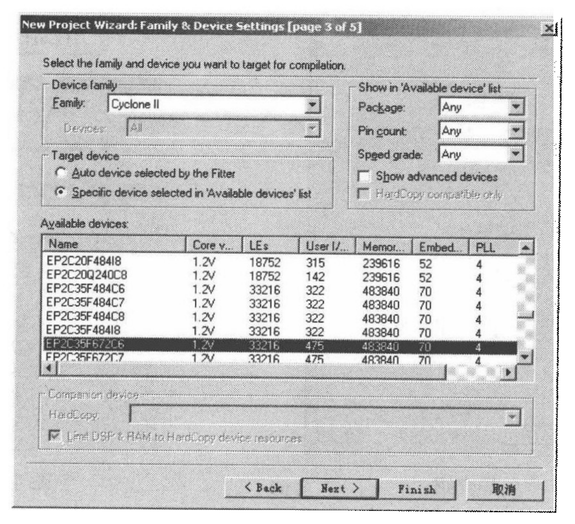


图 4-16　选择设计器件的型号

（3）设定目标器件。在如图 4-16 所示目标器件选择窗口中选择器件的型号（这里的器件型号就是 EDA 实验设备的可编程器件的型号）。在 Available devices 列表中选择开发平台上

的 FPGA 型号，此处选择“EP2C35F672C6”。单击“Next”按钮，进入下一步。

（4）设定 EDA 工具。图 4-17 为设定 EDA 工具的对话框。在这一步，可以为新项目指定第三方综合工具、仿真工具、时间分析工具等专业 EDA 工具。

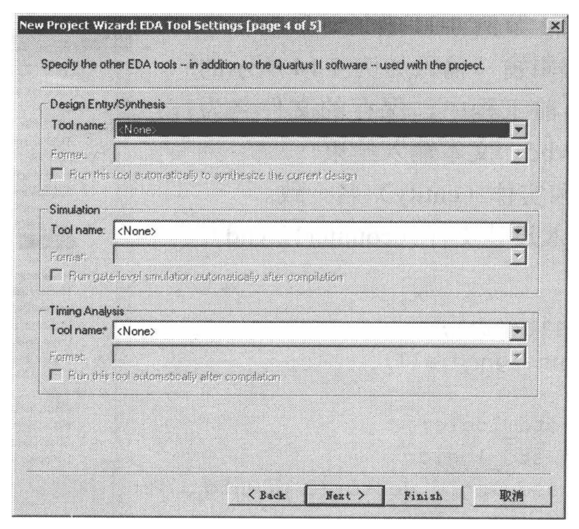


图 4-17　选择仿真器和综合器类型

如果都选默认的“NONE”，表示都选 Quartus II 中自带的仿真器和综合器。在此都选择默认项“NONE”。

（5）工程报告。单击“Next”按钮后，即弹出如图 4-18 所示的工程设置统计窗口，在该窗口内列出了此项工程相关设置情况。最后单击“Finish”按钮，完成新建工程的建立。

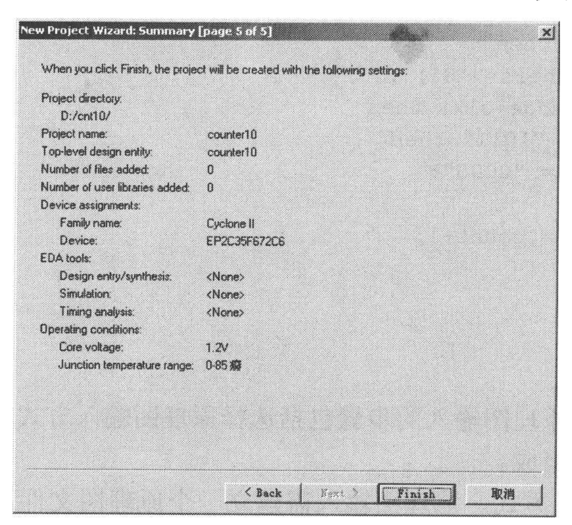


图 4-18　新建工程信息汇总

建立工程后，可以使用“Assignments/Setting”对话框来更改工程中的设计文件、目标器件、仿真综合工具等信息。

2．设计输入

开始设计模 12 计数器，原理图输入所需要的就是“绘制”一张“电路图”，并加入到当

前工程中。VHDL 文本输入所需的就是将设计好的 VHDL 程序输入并加入到当前工程中。

（1）VHDL 文本输入。执行菜单命令“File/New”，打开如图 4-19 所示对话框。

在设计文件（Design File）中选择 VHDL File，单击“OK”按钮，打开文本编辑窗口。在文本编辑窗口中将 4.2.4 节例 4-11 普通计数器程序代码稍作修改，在文本编辑窗口输入如例 4-14 所示的 VHDL 程序，保存到当前工程中，保存的文件名为“counter12”，后缀名为.vhd。文本输入结束。

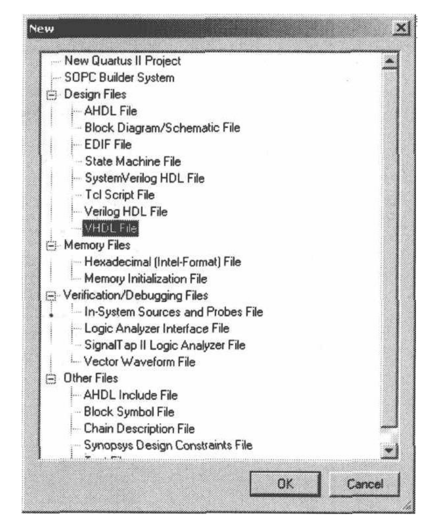


图 4-19　新建文件对话框

注意：文件名必须和实体（entity）名一致。

【例 4-14】 编辑输入设计文件（counter12.vhd）。

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity counter12 is
    Port ( clk : in std_logic;
          reset_n: in std_logic;
          dout : out std_logic_vector(3 downto
0));
end counter12;
architecture Behavioral of counter12 is
   signal count : std_logic_vector(3 downto 0);
begin
    dout <= count;
    process(clk,reset_n)
    begin
       if reset_n='0'then
         count <= (others=>'0') ;
       elsif rising_edge(clk) then
           if count = "1011" then
               count <= "0000";
           else
               count <= count+1;
           end if;
      end if;
    end process;
end Behavioral;
```

（2）原理图输入。原理图输入的步骤包括选择原理图输入方式、放置器件符号、连线、命名引脚和保存文件等组成。

① 选择原理图输入方式。原理图输入需建立一个原理图文件。在新建文件对话框（图 4-19 所示“新建文件”对话框）中，设计者需要选择“Block Diagram/Schematic File”选项，进行原理图的设计，单击“OK”按钮。工作区中弹出空白的图纸——Block1.bdf 文件，并在图纸左侧自动打开绘图工具栏，如图 4-20 所示。

② 放置器件符号。双击原理图的空白处，弹出“元件选择”对话框，如图 4-21 所示。单击原理图输入工具栏中的按钮或者单击鼠标右键，弹出一个选择对话框，选择此框中“Insert”的“Symbol as Block…”项，也可以弹出元件选择对话框。

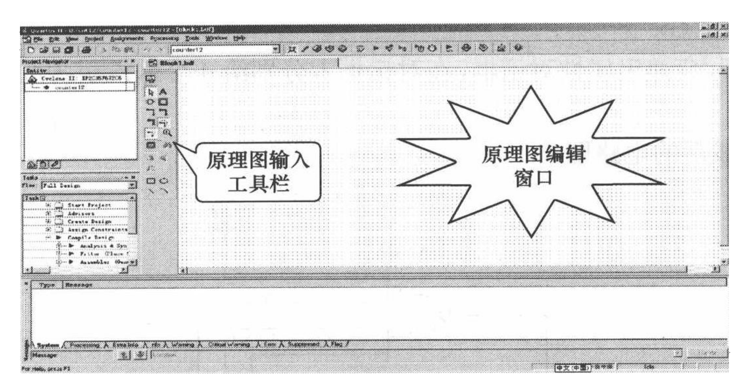


图 4-20　图形编辑文件窗口

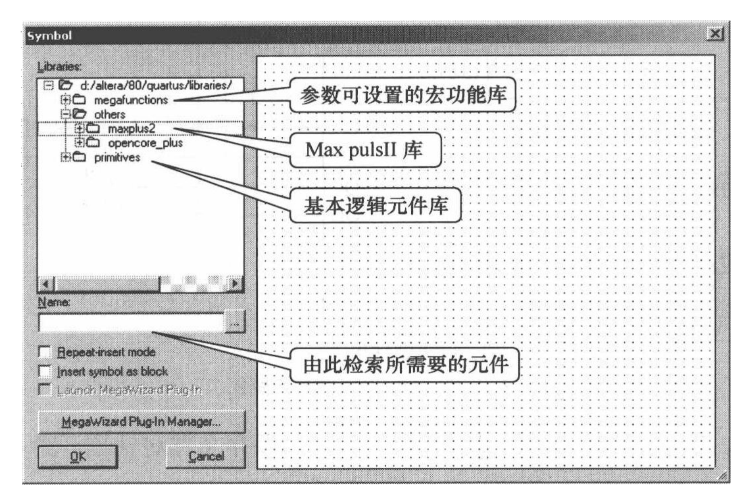


图 4-21　元件选择对话框

在 Name 栏目中输入 74161，就得到一个 4 位二进制计数器。单击“OK”按钮，将其放到原理图的适当位置。重复操作，在 Name 栏目中输入 nand4 将得到一个 4 输入的与非门。单击“OK”按钮，将其放入原理图。用同样的方法，在 Name 栏中分别输入“input”、“output”、“Vcc”和“gnd”，分别得到输入引脚、输出引脚、逻辑高电平和逻辑低电平。

输入元件的另一种方法是双击原理图的任一空白处，在弹出的元件对话框中拖动 Libraries 窗口右边的滑条，类似文件目录结构，可以层层展开库，在展开库中选取所需元件。

“megafunctions”是参数可设置的宏功能库；“others”是 Max pulsII 老式宏函数库，包括加法器、编码器、译码器、计数器以及移位寄存器等 74 系列器件；“primitives”是基本逻辑元件库，包括缓冲器和基本逻辑门，如门电路、触发器、电源、输入和输出等。

在原理图编辑窗口选择某一元件，同时按住 Ctrl 键，拖动即可复制该元件。也可以通过右键菜单的 Copy 命令复制得到。

通过原理图编辑工具栏中的 [图标] 可以调整原理图符号的方向和角度。

③ 连线。将鼠标移到元件的引脚上，鼠标会变成“十”字形状。单击左键，拖动鼠标，就会有导线引出。根据要实现的逻辑，连好各元件的引脚。

④ 命名引脚。双击输入引脚，会弹出属性对话框，在对话框上可更改引脚的名字。在此

给输入引脚取名为“clk”。

双击输出引脚，会弹出属性对话框，在对话框上分别给4个输入引脚取名为“QA”、“QB”、“QC”、“QD”。

图4-22为完成后的电路图。

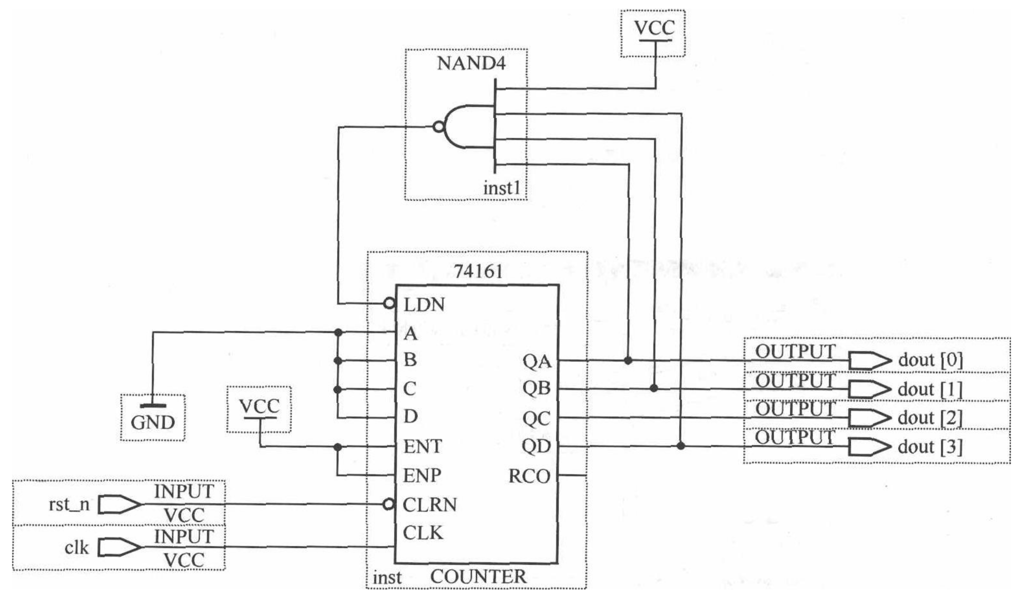


图4-22　模12计数器电路图

⑤ 保存文件。单击工具栏上的保存按钮，弹出保存文件对话框，输入文件名，单击“保存”按钮即可。

3. 分析综合

设计输入完成后，可以先检查设计文件的语法错误和逻辑错误。选择 Processing/Start/Start Analysis & Synthesis 或分析并综合 VHDL 设计。分析综合使用多种算法来减少逻辑门的数量，删除冗余逻辑以及尽可能有效地利用器件体系结构。分析综合阶段检查工程的逻辑完整性和一致性，并检查边界连接和语法错误。在此阶段若有错误，会在 Message 窗口中显示错误信息。

若在此阶段设计文件有语法错误，则按照 Message 窗口中的提示错误进行更改。

4. 电路仿真

仿真一般需要经过建立波形文件、输入信号节点、设置波形参量、编辑输入信号、波形文件存盘、运行仿真器和分析仿真波形等步骤。

Quartus II 支持功能仿真和时序仿真，功能仿真只检验设计项目的逻辑功能，时序仿真则将延时信息也考虑在内，更符合系统的实际工作情况。

为了验证设计的正确性，接下来对程序进行功能仿真（由于时序仿真耗时较长，复杂设计和高频设计时，建议功能功能仿真后，进行时序分析，设计优化，最后通过嵌入式逻辑分析仪进行信号的分析）。

（1）建立矢量波形文件。执行“File/New”菜单命令，在“Verification/Debugging File”下建立“Vector Waveform File”，如图 4-23 所示。单击“OK”按钮，进入矢量波形编辑器窗口，如图 4-24 所示。

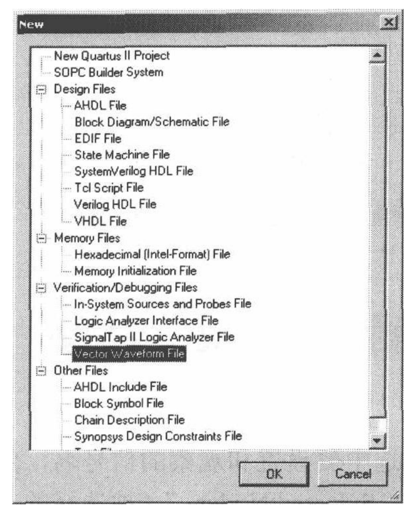


图 4-23　新建矢量波形文件

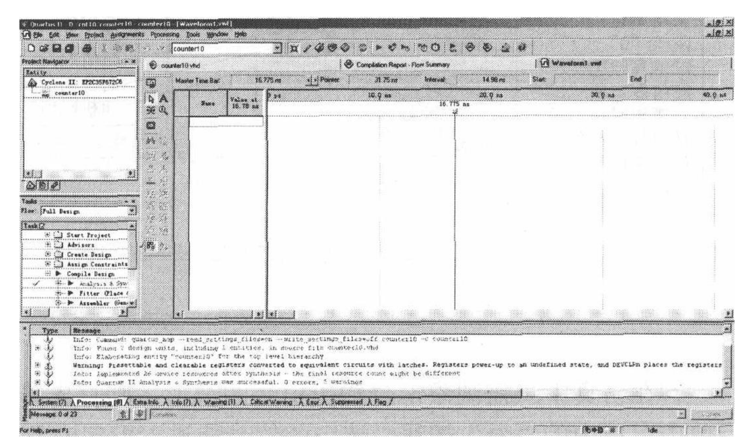


图 4-24　矢量波形编辑器窗口

使用“File/Save as”菜单命令将文件保存为“counter12.vwf”。

（2）将输入端口节点和需要观察的输出端口节点加入到波形文件中。执行“Edit/Insert Node or Bus...”菜单命令，或在信号窗口双击，打开如图 4-25 所示的对话框。

单击“Node Finder...”按钮，在 Node Finder 窗口中选择 Filter 为“pins：all”，单击“List”按钮，这时在窗口左边“Nodes Founder”框中列出该设计项目的全部信号节点。在仿真中需要设置全部输入信号和需观察的输出信号的波形。单击窗口中的“>>”按钮，将信号全部加入，结果如图 4-26 所示。单击“OK”按钮，确认。回到“Insert Node or Bus”对话框，单击“OK”按钮，出现如图 4-27 所示的波形编辑窗口。

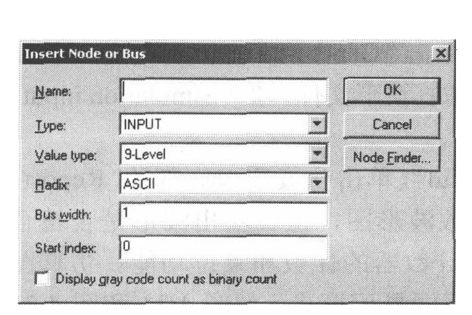


图 4-25　插入节点/总线对话框

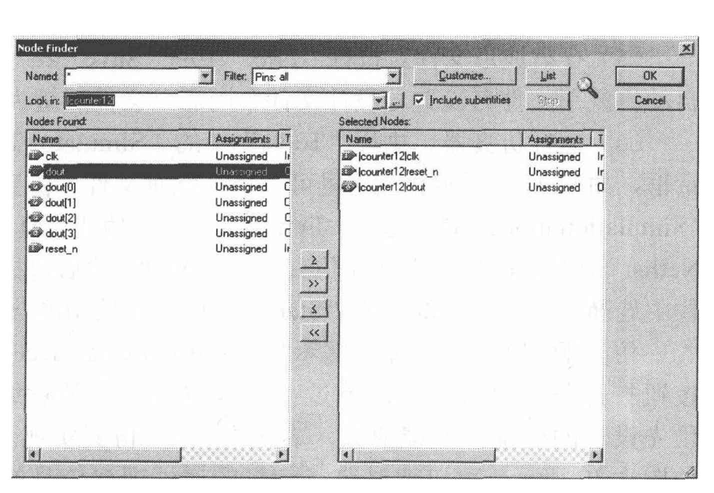


图 4-26　节点发现者对话框

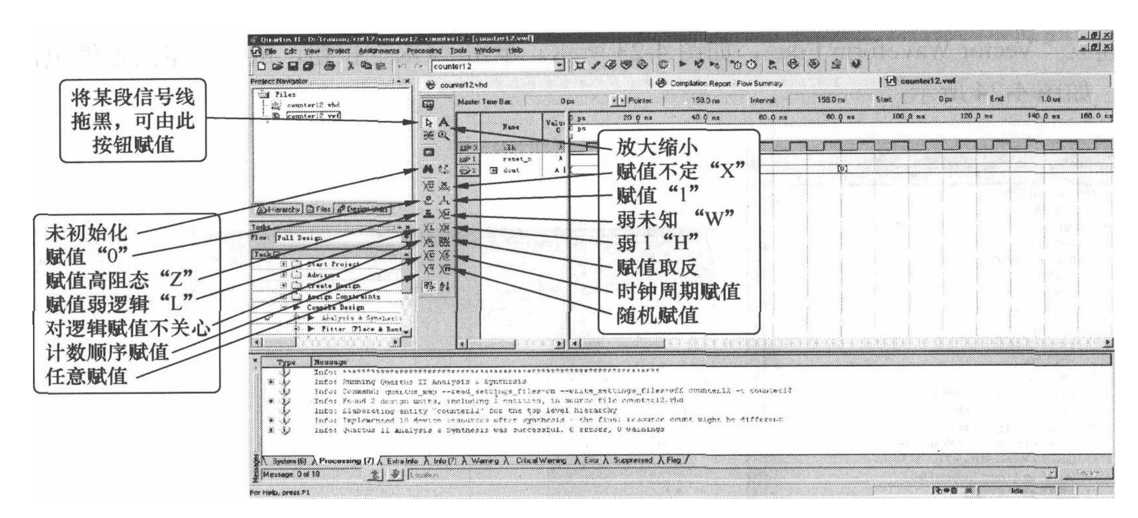


图 4-27 波形编辑窗口

若在仿真中只需观察部分输出信号的波形，则首先单击选中要设置和观察的信号名，然后单击窗口中的“≥”按钮，选中的信号即进入窗口右侧的“Selected Nodes:”(被选择的节点)框中。如果需要删除“Selected Nodes:”框中的节点信号，可用鼠标将其选中，然后单击窗口中的“≤”按钮。

(3) 设置仿真时间。Quartus II 默认的仿真时间域是 1μs，为了使仿真时间轴设置在一个合理的区域上，执行“Edit/End Time”菜单命令，在弹出窗口中的“End Time”窗口输入适合的仿真时间域，单击“OK”，结束设置。

(4) 编辑输入信号。编辑 clk 信号的波形。先选中 clk 信号，然后单击“时钟信号”按钮，采用周期为 10ns，占空比（指高电平在一个周期内所占的时间比率）为 50%的时钟信号，如图 4-28 所示，单击“OK”按钮以确定。在该对话框中可设置时钟信号的周期、相位和占空比。

同样，利用必要的功能键，如图 4-27 所示，为 reset_n 设置必要的逻辑。

(5) 存盘波形文件。执行“File”中的“Save”菜单命令。在波形文件存盘操作中，系统自动将波形文件名设置与设计文件名同名，文件后缀为.vwf。

(6) 运行仿真器。执行“Tools”中的“Simulator Tool”菜单命令，弹出“仿真工具”对话框，如图 4-29 所示。在这里可以选择激励文件、仿真模式（功能仿真或时序仿真）等。在“Simulation mode”栏中选择“Functional”，即功能仿真。单击“Generate Functional Simulation Netlist”按钮，产生用于功能仿真的网表文件。然后选择仿真波形文件，在“Simulation input”栏中选择。单击对话框中的“Start”按钮，开始功能仿真。

仿真顺利通过，系统会提示“Simulation was successful”。单击仿真窗口底部的“Report”按钮打开仿真报告。可以通过“缩放按钮”随意缩放波形图，检查输出波形是否符合预先设计的要求（左键放大，右键缩小）。仿真实现了计数器的计数和复位功能。仿真图如图 4-30 所示。可以通过节点信号的属性观察信号的不同显示形式，如图 4-31 和图 4-32 所示。

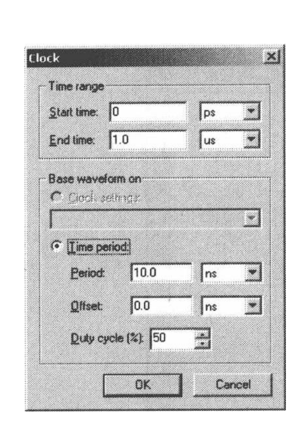


图 4-28 Clock 信号编辑对话框

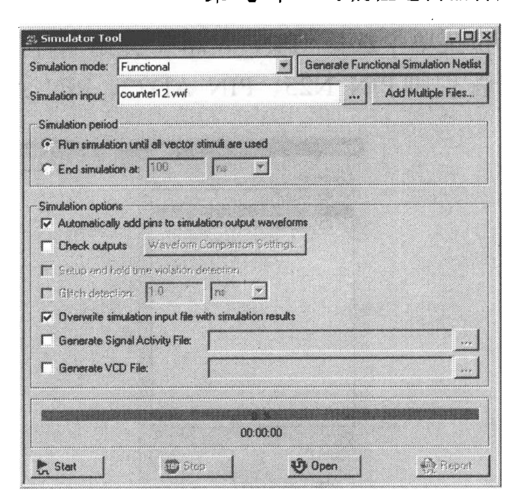


图 4-29 仿真工具对话框

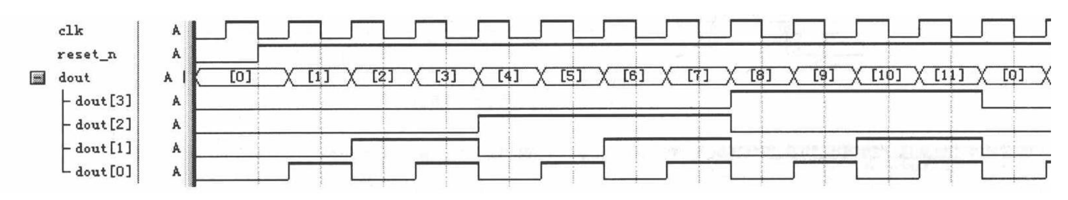


图 4-30 仿真波形图

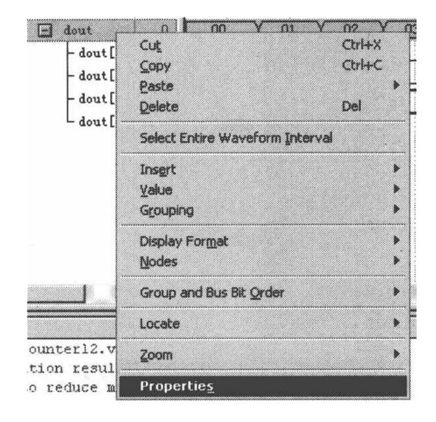


图 4-31 信号属性

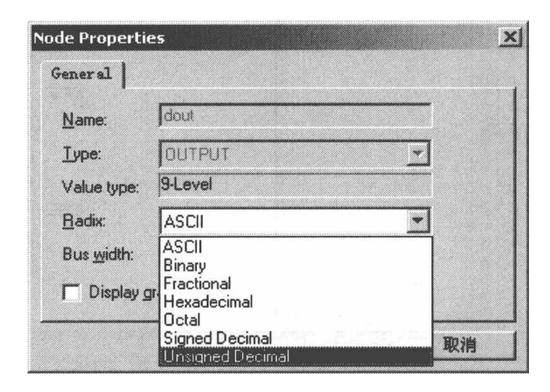


图 4-32 通过节点属性观察信号

5. 创建模块

单击“File/Creat/Updat/Creat Symbol Files for Current File”菜单命令，便可创建当前项目的模块，并添加到模块库中，且可作为底层文件供以后设计调用，如图 4-33 所示。

6. 分配引脚

分配引脚是为了对设计进行硬件测试和实际使用，将输入输出锁定在器件的实际管脚上。

执行“Assignments /Pins”菜单命令，弹出如图 4-34 所示的引脚规划器，在引脚规划器下部的 Filter 栏中选择“Pin：All”，窗格中列出了所有引脚。节点 clk、rst_en、dout[3]、dout[2]、dout[1]、dout[0]对应的 location 一栏都是空的，说明没有分配引脚，单击节点 clk 的 location

栏，在下拉菜单中选择“PIN_W26”，并用同样的方法分别为 rst_en、dout[3]、dout[2]、dout[1]、dout[0]分配引脚 PIN_N25、PIN_AD22、PIN_AC22、PIN_AB21、PIN_AF23。

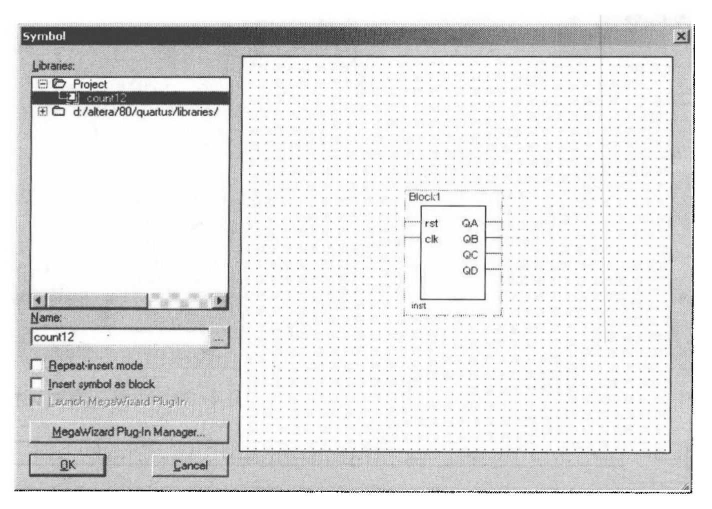


图 4-33　创建原理图模块

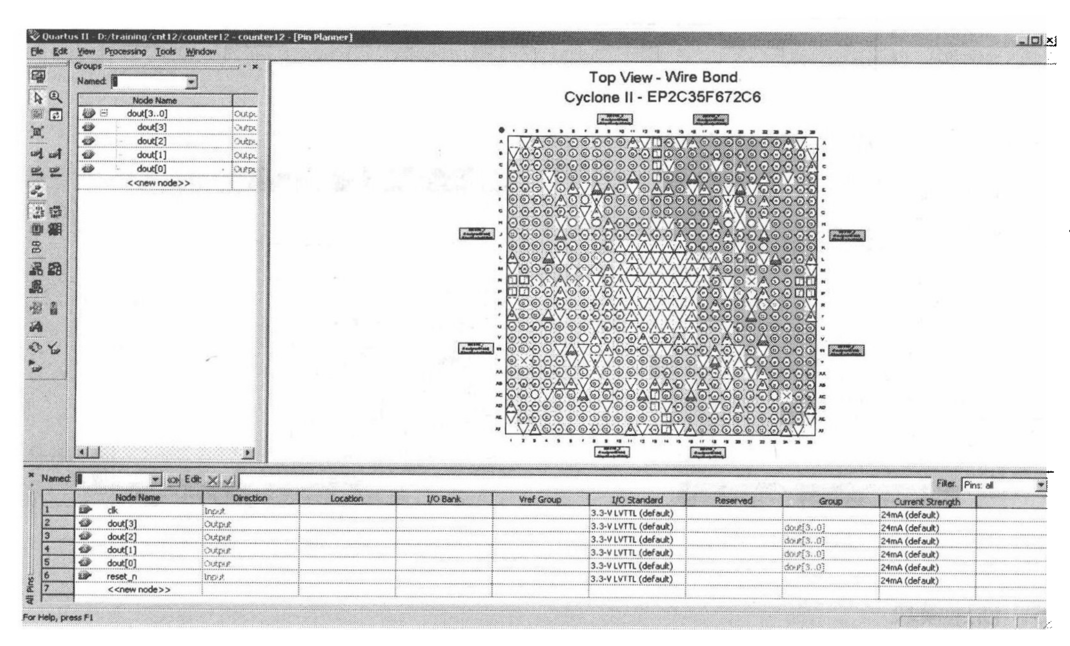


图 4-34　引脚规划器窗口

最后存储这些引脚锁定的信息后，必须对锁定引脚后设计文件重新编译（启动 Processing\Start Compilation），以使引脚锁定信息编译到编程下载文件（.sof）中。此后，就可以准备将编译好的 SOF 文件下载到 EDA 实验系统的 FPGA 芯片中了。

7．编译设计文件

在“Tool”菜单中单击“Compiler Tool”命令或▶按钮，启编译窗口，工程开始编译它所包含的设计文件。编译的过程分为分析与综合、适配、编译和时序分析 4 个步骤。

编译过程中的相关信息将在“消息窗口”中出现。编译结束之后，Quartus II 会自动打开 Compilation Report 窗口，如图 4-35 所示。报告表明这个电路占用了 4 个逻辑单元、4 个寄存器和 6 个引脚。用窗口左侧的层次结构可以查看各部分的详细报告。

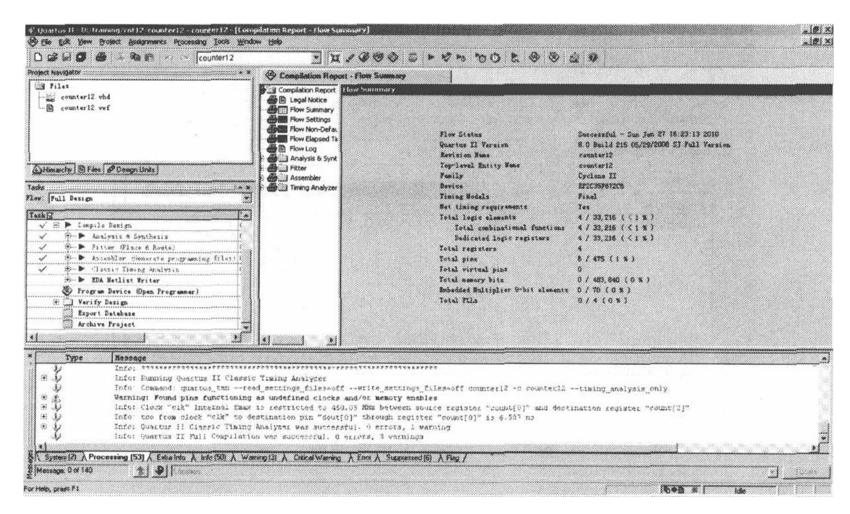


图 4-35　成功编译后的显示

8．编程下载并硬件测试

（1）连接 PC 与目标开发板。Altera FPGA 的早期下载线缆主要有并口的 ByteBlasterMV 和 ByteBlaster II、串口的 MasterBlaster 和 RJ-45 网络接口。近几年来，由于 USB 接口的普及，编程线缆多以 USB 下载线缆居多。

USB Blaster 下载线的驱动安装：将 USB Blaster 下载线缆连入计算机后，首次使用时必须安装驱动。USBBlaster 的驱动程序在 Quartus II 安装目录下（如 c：\altera\80\quartus\drivers\usb-blaster\）。驱动程序的安装与其他 USB 设备驱动程序的安装类似。

（2）配置下载电缆。执行“Tools\ Programmer”菜单命令或者单击工具栏中的按钮，弹出如图 4-36 所示的编程配置下载窗口，在该窗口中，单击“Hardware Setup”按钮，弹出所

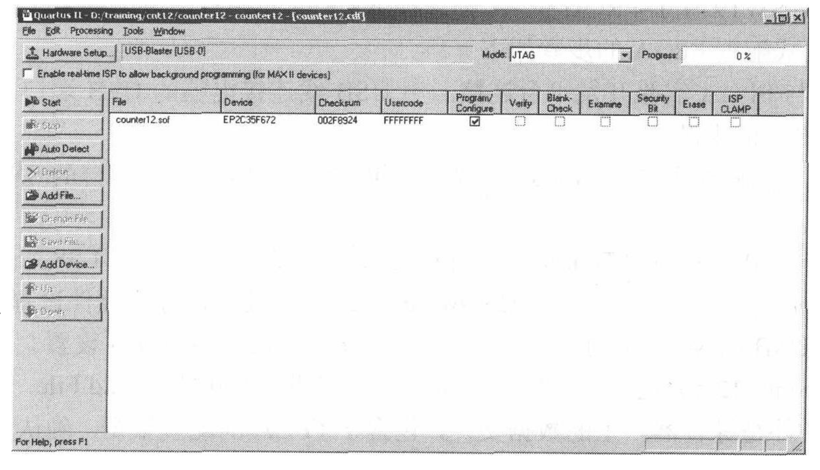


图 4-36　编程配置窗口

示的硬件设置对话框。单击“Hardware Settings”按钮，在 Currently selected Hardware 下拉列表中选择 USB-Blaster [USB-0]选项，如图 4-37 所示。然后单击“Close”按钮，关闭该对话框，完成下载线配置。

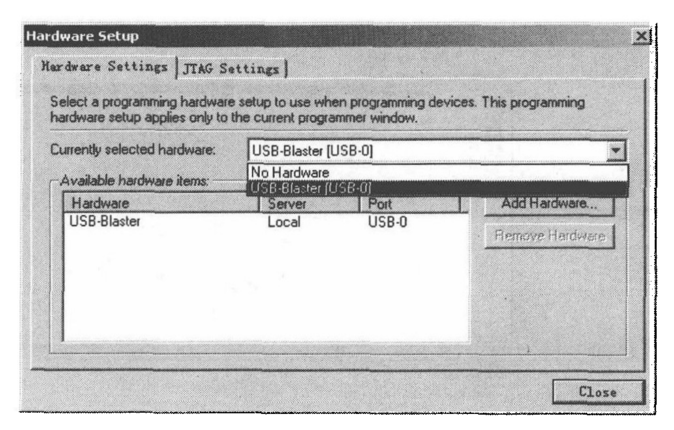


图 4-37　硬件设置对话框

（3）下载模式。USB-Blaster 下载线支持 3 种下载模式：JTAG 模式、AS（Active Serial）模式和 PS（Passive Serial）模式。其中，JTAG 模式是软件默认的下载方式，对应下载的文件格式是.sof。利用此模式下载，配置文件下载到目标 FPGA 中的 SRAM 内，一旦开发板电源关闭，下次使用时需要重新下载，此模式适用于在线调试。AS 和 PS 模式是将配置文件数据下载到目标板的配置芯片中，目标板断开电源，下次使用无需重新下载，AS 模式对应的下载文件为.pof。

下面以 DE2 开发板为例把下载过程简单做一些总结，其他开发板和此下载过程类似。

DE2 平台上嵌入了 USB Blaster 下载组件，可以通过一条 USB 连接线和电脑相连，通过 JTAG 模式和 AS 模式配置 FPGA。JTAG 模式通过 USB Blaster 直接配置 FPGA，但断电后配置内容丢失，再次通电后需用电脑重新对 FPGA 配置；AS 模式通过 USB Blaster 对 DE2 平台上的串行配置器件 EPCS16 进行配置，平台通电后，配置器件 EPCS16 自动配置 FPGA。

也可以通过 DE2 平台上的 SW19 选择配置模式，SW19 置于 RUN 位置，为 JTAG 模式直接配置 FPGA；SW19 置于 PROG 位置，则选择 AS 模式对串行配置器件 EPCS16 进行配置。

用 JTAG 模式配置 FPGA 的步骤如下：

① 将电脑与 FPGA 实验开发平台连接。用 USB 线连接电脑的 USB 端口与 DE2 平台的 J9，打开 DE2 平台的电源。

② 使开发平台硬件连接为 JTAG 模式配置 FPGA。这里将 DE2 平台的 SW19 置于 RUN 位置。

③ 在 Quartus II 中执行“Tool /Programmer”菜单命令，打开如图 4-38 所示的窗口。

④ 如果显示没有硬件，则单击“Hardware Setup…”按钮，打开硬件设置窗口。

⑤ 双击“USB Blaster”按钮，然后单击“Close”按钮，完成硬件设置。

⑥ 此时 counter12.sof 已经在文件列表中了，如果没有则单击“Add File…”按钮，添加该文件，这个文件是综合器产生的数据文件，包含了 FPGA 的配置数据。确认 Device 项列出的器件是 EP2C35F672，选中“Program/Configure”选项，如图 4-38 所示。

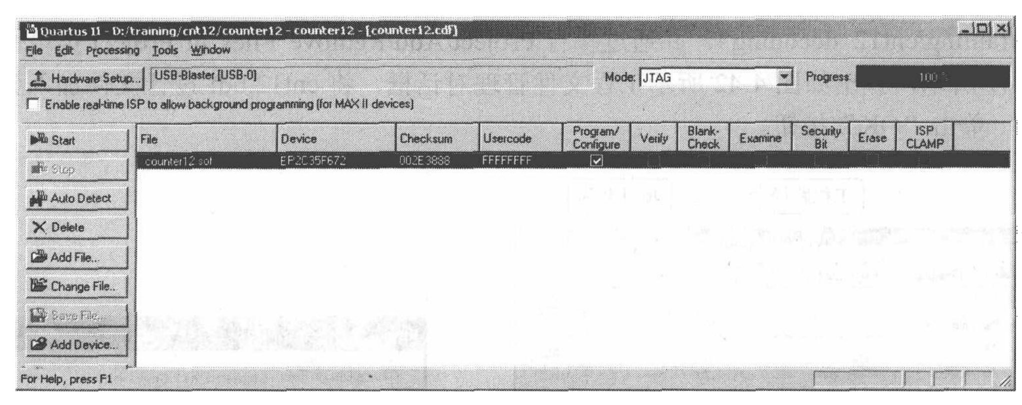


图 4-38　更新后的编程窗口

⑦ 单击下载标符“Start”按钮，即进入对目标器件 FPGA 的配置下载操作。配置完成后，DE2 板上的配置完成指示二极管 GOOD 变亮。如果 Quartus II 提示错误，则检查电源及连接电缆。

为了使 FPGA 在通电后仍然保持原有的配置文件，并能正常工作，必须将配置文件写入专用的配置芯片 EPCS16 中。EPCS16 是 Cyclone 系列器件的专用配置器件，Flash 存储结构，编程次数为 10 万次左右。编程模式为 Active Serial 模式。对应下载的文件为 counter12.pof。

4.5.2　原理图、文本混合输入方法

上一节讲述了 QuartusII 软件的基本使用方法，本节将上节内容进行延伸，通过对十二进制计数器进行译码，并在数码管上显示计数值，来介绍层次化的设计方法以及原理图、文本混合输入方法。

设计分析：上节设计的十二进制计数器的计数范围是“0000”-“1011”,，也就是十进制的 0-11。在进行译码之前必须进行二进制到十进制的转换，即十进制“10”、“11”要用两位数码管来显示。

由以上分析，设计模块应包括十二进制计数器模块、B-BCD（2-10 进制转换）转换模块、译码模块。当然，若在写程序时考虑到输出需要 BCD 码，可以直接设计 BCD 码输出的十二进制计数器。

1．使用“New Project Wizard”命令新建一个工程

首先建立工程文件夹，如：D:/training/cnt12_decoding。在如图 4-39 所示的文件中填入工程项目名称及顶层文件。

2．设计输入、编译、仿真

Quartus II 的文本编辑输入法与图形输入法的设计步骤基本相同。只是在第二步设计输入文件时选择 VHDL File，如图 4-40 所示，注意设计文件保存时，保存的文件名必须和所定义的实体名相同。其他设计步骤和图形输入法相同。

（1）底层设计——2 进制计数器：cnt12。

采用上节的原理图输入，将上节工程中的原理图文件 cnt12.bdf 拷贝到当前工程文件夹下

（如 D:/training/cnt12_decoding），并通过执行 Project/Add/Remove Files in Project 菜单命令，如图 4-41 所示，弹出如图 4-42 所示工程文件管理对话框，将 cnt12.bdf 设计文件加入到当前工程中，单击“OK”按钮。

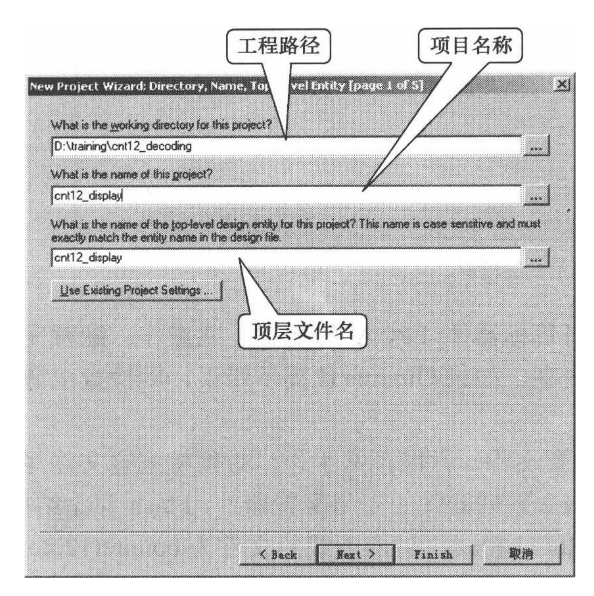


图 4-39 新建工程设置对话框

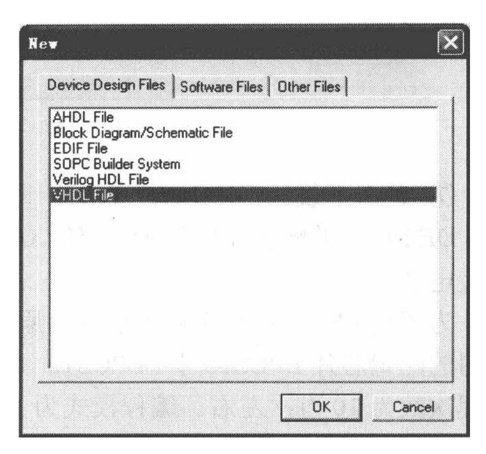


图 4-40 选择描述文件类型

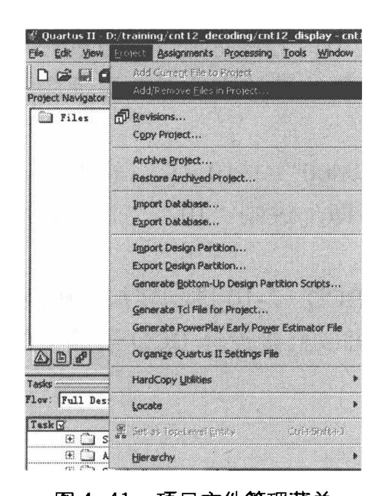


图 4-41 项目文件管理菜单

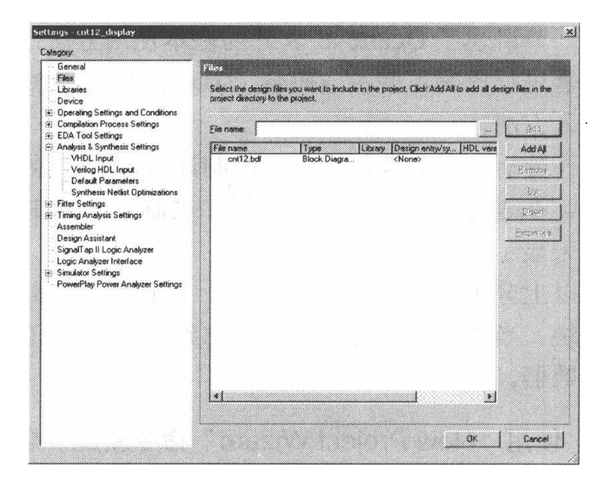


图 4-42 工程文件管理对话框

单击“File/Creat/Updat/Creat Symbol Files for Current File”，便可创建当前项目的模块，并添加到模块库中，可作为底层文件供以后设计调用，如图 4-15 所示。

（2）底层设计——2-10 进制转换：B_BCD.VHD。

在当前工程下，新建 VHDL 文本输入文件，完成 B_BCD 的 VHDL 设计输入，保存为 B_BCD.VHD，并将其设置为顶层实体（通过执行“Project/Set as Top Level Entity”菜单命令）。然后进行编译，功能仿真。步骤如原理图输入法。如编译、仿真成功，则 B_BCD 模块设计结束。

单击“File/Creat/Updat/Creat Symbol Files for Current File”，创建当前项目的模块，并

添加到模块库中，可作为底层文件供以后设计调用。

2-10进制转换的逻辑描述源文件（底层文件）：

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity B_BCD is
 port(clk:in std_logic;
    din0,din1,din2,din3:in std_logic;
     y0,y1:out std_logic_vector(3 downto 0));
end B_BCD;
architecture behav of B_BCD is
 signal mid_in:std_logic_vector(7 downto 0);
 signal din:std_logic_vector(3 downto 0);
 begin
    din<=din3&din2&din1&din0;                         --并置运算
    process(din)
    begin
    if (din>"1001") then
        mid_in<="0000"&din+"00000110";
        y0<=mid_in(3 downto 0);
        y1<=mid_in(7 downto 4);
    else y0<=din;
        y1<="0000";
    end if;
    end process;
end architecture  behav;
```

以上程序设计思路：二进制转换为BCD码，如果二进制大于9，则通过加6修正，小于9，则直接输出。详细请参考数字电路码制转换的相应章节。

（3）底层设计——七段译码器模块decoder.vhd。

方法和步骤如2-10进制转换模块的设计，如果编译、仿真成功，则七段译码器模块设计结束。

单击“File/Creat/Updat/Creat Symbol Files for Current File”，创建当前项目的模块，并添加到模块库中，可作为底层文件供以后设计调用。

译码模块的逻辑描述源文件（底层文件）：（该程序的LED七段码）

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity decoder is
   Port (din_a:in std_logic;
         din_b:in std_logic;
        din_c:in std_logic;
        din_d:in std_logic;                             --四位二进制码输入
         dout:out std_logic_vector(6 downto 0) );      --输出LED七段码
end decoder;
architecture Behavioral of decoder is
signal din:std_logic_vector(3 downto 0);
begin
```

```
din<=(din_d&din_c&din_b&din_a);                          --并置运算
process(din)
    begin
        case din is
            when "0000" => dout<="0000001";--0 该程序的 LED 七段码为共阳极
            when "0001" => dout<="1001111";--1
            when "0010" => dout<="0010010";--2
            when "0011" => dout<="0000110";--3
            when "0100" => dout<="1001100";--4
            when "0101" => dout<="0100100";--5
            when "0110" => dout<="0100000";--6
            when "0111" => dout<="0001111";--7
            when "1000" => dout<="0000000";--8
            when "1001" => dout<="0000100";--9
            when others => dout<="1111111";
        end case;
end process;
end Behavioral;
```

（4）顶层设计——设计计数译码显示。

本例中，通过原理图输入作为顶层文件。新建原理图文件，把以上设计的各模块通过添加器件的方法加到顶层原理图中，如图 4-43 所示。设计好的顶层原理图如图 4-44 所示，保存顶层文件为 cnt12_display.bdf（.bdf 为原理图后缀名），通过执行“Project|Set as Top Level Entity”菜单命令将其设置为顶层实体。

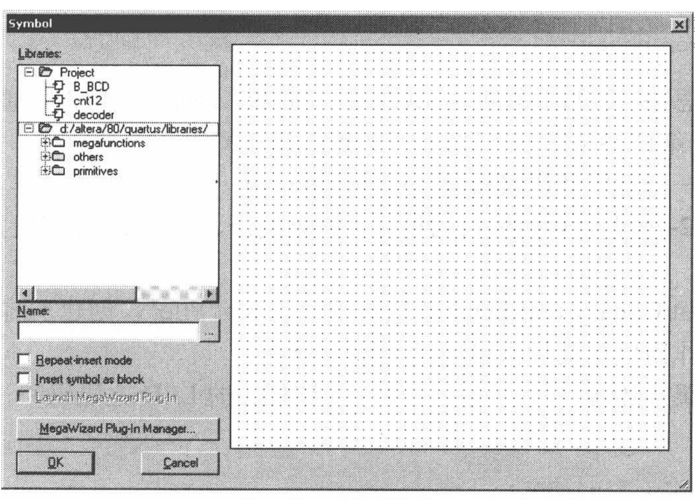


图 4-43　添加器件窗口

由于 B_BCD 模块输出采用的是标准逻辑矢量（std_logic_vector）输出，而 decoder 模块采用的是标准逻辑（std_logic）输出，在顶层连线时带来了一些问题，在此例中通过网络标号来对不匹配的端口进行连接。

① 要通过 net（网络标号）来连接总线和分立引脚，总线编号规则是 din[3..0]，他包含 3 个信号线 din[3]、din[2]、din[1]和 din [0]。与此总线相连的引脚线编号是 din[3]、din[2]、din[1]和 din [0]。

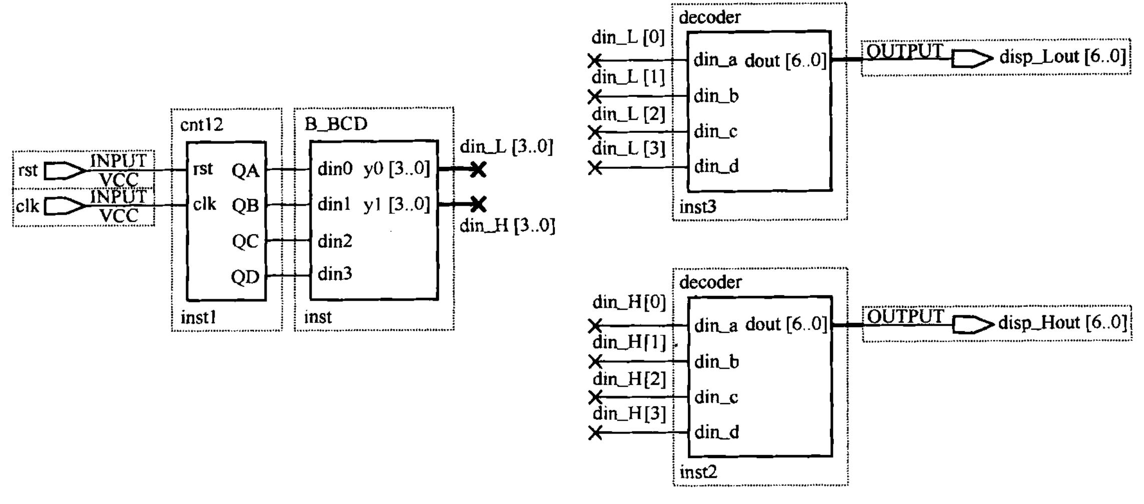


图 4-44　顶层原理图

② 网络标号添加：选中连线后，单击鼠标右键，选择属性选项（Properties），弹出“总线属性”对话框，输入网络标号。详见图 4-45 和图 4-46。

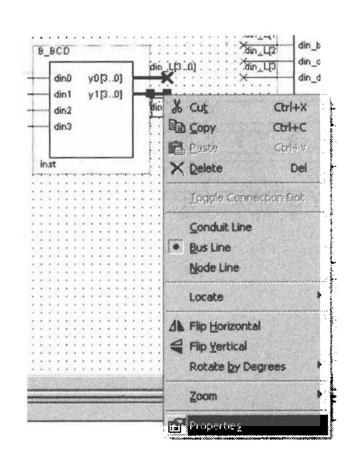


图 4-45　总线属性

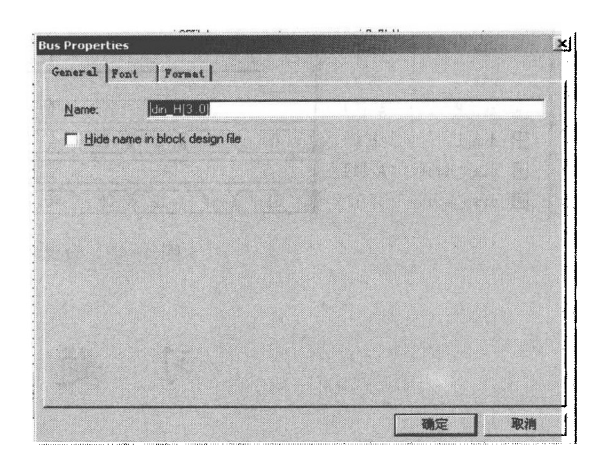


图 4-46　更改网络标号对话框

③ 为了便于调试，增加了两个内部信号的测试端口，din_H[3..0]和 din_L[3..0]，如图 4-47 所示。

至此，设计输入结束，对顶层文件仿真。仿真图如图 4-48 所示，由图可见设计正确，实现了模 12 的计数和 BCD 译码。

3．分配引脚、编译并编程下载测试

对本项目分配引脚并在实验板上下载验证其功能，参考 4.5.1 节，在此不再叙述。

本节分别使用 Quartus II 软件的原理图、VHDL 语言以及混合模式输入 3 种设计 FPGA 数字系统的方法进行了简单的项目设计。要达到熟练运用这些方法，还需要长期的练习和不断地总结经验。

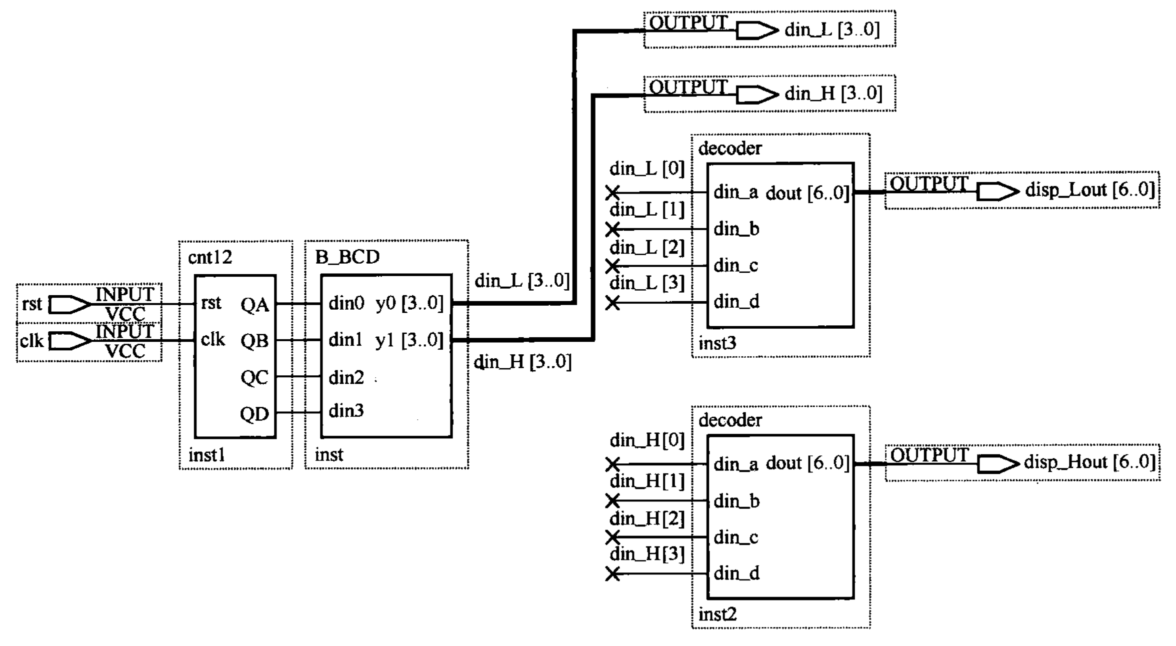


图 4-47 加了测试信号端口的顶层文件

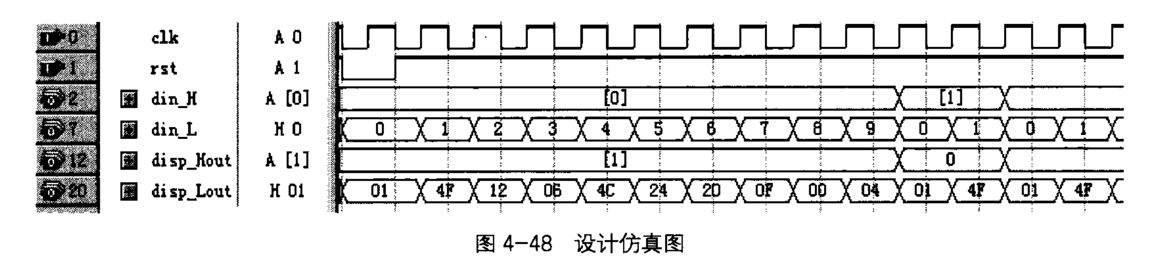


图 4-48 设计仿真图

习 题 四

1．简述目前常用的可编程逻辑器件的分类及相应特点。

2．VHDL 中，设计实体由哪几部分组成？各部分的功能是什么？

3．VHDL 中，为什么引入库和程序包？库和程序包的功能是什么？

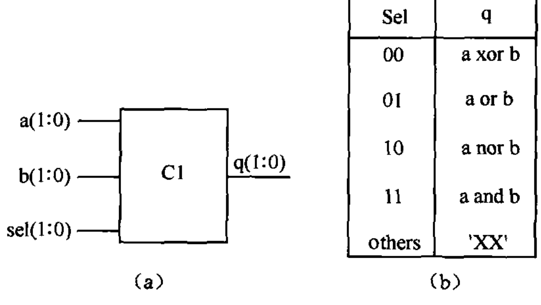
4．VHDL 中，常用的库和程序包有哪些？

5．VHDL 的结构体描述语句有哪些？

6．术语“功能仿真”、“时序仿真”表示什么意思？为什么对一个设计在一般情况下可以只做功能仿真而不再做时序仿真？

7．设计一个元件，外部接口如图 4-49（a）所示，图左侧为输入信号，右侧为输出信号。该元件应当有如图 4-49（b）的行为。

（1）用 if 语句。（2）用 case 语句。（3）用 when else 语句。



(a) (b)

图 4-49 外部接口及元件行为图

8. 在处理组合逻辑电路时初学者经常在不需要锁存器（LATCH）时，意外生成了锁存器。请判断下列 2 个进程是否产生了锁存器，如产生了锁存器，如何修改？

(1)

```
Process (A, B, C, Cntr)
begin
    if  (Cntr = '1' ) then
       A <= '1'
       B <= '1'
    else
       C <= '1'
    end if;
end process ;
```

(2)

```
process  (A, B, C, D, S)
    begin
       A <= '0' ;
       B <= '0' ;
       C <= '0' ;
       D <= '0' ;
    case S is
       when "00" =>
          A <= '1' ;
       when "01" =>
          B <= '1' ;
       when "10" =>
          C <= '1' ;
       when "11" =>
          D <= '1' ;
       when others =>
          null ;
    end case ;
end process ;
```

第5章 集成电路制造工艺与专用集成电路设计

本章要点

- 集成电路主要的制造工艺
- CMOS 基本单元电路
- 专业集成电路的设计过程
- 专业集成电路的 EDA 技术

5.1 集成电路制造工艺简介

为了从整体上去学习集成电路设计的方法和过程，有必要对集成电路制造的工艺做些简单的介绍。集成电路的制造工艺包括衬底外延生长、掩膜制版、光刻、掺杂、绝缘层、金属层形成等，下面介绍几个主要的工艺。

1．外延生长

半导体工艺流程中的衬底是抛光过的晶圆（Wafer）基片，直径在 50mm～200mm 之间，厚度约几百微米。大多数的器件都是做在经过外延生长的衬底上。“外延”一词系指在单晶衬底上生长一层新单晶的技术。新生单晶层的晶向取决于衬底，由衬底向外延伸而成，故称为“外延层”。当衬底与外延成为同种材料时称为同质外延，例如，在硅衬底上外延硅。这时外延的目的是形成具有不同掺杂种类及浓度的晶体层，因而它可以具有不同性能。当两者材料相异时称异质外延，例如，在硅衬底上外延锗。异质外延用来形成各种异质结构的器件，如异质结晶体管（HBT）。

外延生长之所以重要，在于外延层中的杂质浓度可以方便地通过控制反应气流中的杂质含量加以调节，而不依赖于衬底中的杂质种类与掺杂水平。这样，就能在外延层与衬底之间形成 PN 结。这种 PN 结与扩散结不同，它并不是通过杂质的补偿作用形成的，因而，其杂质分布可接近于理想的突变结。双级型集成电路元器件间的间隔问题可通过外延与间隔扩散技术相结合而获得解决。外延技术还可以解决高频功率器件的击穿电压与集电极串联电阻对集电极电阻率要求之间的矛盾。掺杂较少的外延层保证了较高的击穿电压，高掺杂的衬底降低了集电极的串联电路。

不同的外延工艺可制造出不同的材料系统。目前常用的有气相外延生长（VPE：Vapor Phase Epitaxy）、金属有机物气相外延生长（MOVPE：Metal-Organic Vapor Phase Epitaxy）、和分子束外延生长（MBE：Molecular Beam Epitaxy）。

2. 掩膜的制版工艺

在外延的晶圆上，由工艺工程师进行集成电路制造的一系列工序，而电路设计工程师为集成电路的制造设计出了一系列物理定义的抽象表达——版图。在计算机及辅助设计软件中设计的集成电路版图要送到工艺线上生产时，必须经过一个重要的中间环节——制版。

制版就是要产生一套分层的版图掩膜，为将来进行图形转移，即将设计的版图移到晶圆上去做准备。

在集成电路的版图设计完成以后，就会得到一组标准的制版数据，将这组数据给制版设备，制版设备根据数据将设计的版图分层地转移到掩膜版上（掩膜版为涂有感光材料的优质玻璃板），这个过程叫初缩。在获得分层的初缩版后，再通过分步重复技术，在最终的掩膜版上产生具有一定行数和列数的重复图形阵列，这样在将来制作的每一片晶圆上将有若干的集成电路芯片。通过这样的制版过程，就可产生若干块的集成电路分层掩膜版。集成电路加工过程的复杂程度和制作周期在很大程度上与掩膜版的多少有关。

3. 光刻

光刻是集成电路加工过程中的重要工序，作用是把掩膜版上的图形转换成晶圆上的器件结构。光刻对集成电路图形结构的形成，如各层薄膜的图形及掺杂区域等，均起着决定性的作用。通常可用光刻次数及所需掩膜的个数来表示某生产工艺的难易程度。集成电路的特征尺寸是否能够进一步减小，也与光刻技术的进一步发展有密切的关系。通常人们用特征尺寸来评价一个集成电路生产的技术水平。

超大规模集成电路对光刻的基本要求包括：高分辨率、高灵敏度、精密的套刻对准、尺寸硅片上的加工、低缺陷等。光刻的步骤一般包括涂光刻胶、曝光、显影与后烘以及刻蚀 4 个步骤。

4. 掺杂

掺杂的目的是制作 N 型或 P 型半导体区域，以构成各种器件结构。掺杂工艺的基本思想就是通过某种技术措施，将一定浓度的诸如硼等三价元素或诸如磷、砷等五价元素渗入半导体衬底。通过掺杂，原材料的部分原子被杂质原子代替。若在 N 型衬底上掺杂磷或在 P 型衬底上掺杂硼，均可提高原衬底表面杂质的浓度；若在 N 型衬底上掺杂硼或在 P 型衬底上掺杂磷，可以降低原衬底表面杂质所产生的多数载流子的浓度，或将原衬底反型。

掺杂一般分为热扩散法掺杂和离子注入法掺杂，这里就不作详细说明，读者可以根据自己兴趣查找相关资料。

5. 绝缘层的形成

在集成电路里的工艺里，导体和绝缘体是互补而又相对的。在晶体外层，绝缘层不仅起到电隔离的作用，还可以起到防止湿气侵扰、化学玷污和机械划伤的保护作用。因此，晶圆

经过外延生长之后，首先在其表面形成一层绝缘层，为以后或长时间内的进一步加工做好准备。此外，在制作器件时，必须同时制作器件之间、有源层和导线层之间的绝缘层，在 MOS 器件里，栅极与沟道之间的绝缘更是必不可少的。

在所有的 Si 工艺中，SiO_2 层既可以作为阻止离子注入及热扩散的掩膜，又可被广泛应用于制作绝缘层。这时绝缘层的形成技术主要就是硅的氧化技术。人们已经研究出多种用于形成氧化硅的技术，如热氧化、湿法阳极氧化、气相技术、等离子阳极氧化或等离子氧化。

6. 金属层的形成

金属层的形成主要采用物理气相沉积法（Physical Vapor Deposition，简称 PVD）技术。在半导体工艺发展过程中，主要的 PVD 技术有蒸馏和溅镀两种。

金属层的功能有 3 种，分别是：形成器件本身的接触线、形成器件间的互联线以及形成焊接。

5.2 CMOS 基本单元电路

根据晶体管的性质，集成电路可分为 TTL（Transister-Transister-Logic ）和 COMS（Complementary Metal-Oxide Semiconductor）两大类。CMOS 以功耗低的优势，成为目前应用最广泛的集成电路。这里重点介绍 COMS 基本单元电路。

互补金属氧化物半导体（CMOS）是一种大规模应用于集成电路（IC）芯片的原料。集成电路是将多个单独的集成电路集成到一个电路中，产生一个十分紧凑的器件。

CMOS 电路由绝缘场效应晶体管组成。由于只有一种载流子，因而是一种单极型晶体管集成电路。其基本结构是一个 N 沟道 MOS 管和一个 P 沟道 MOS 管。由于两管栅极工作电压极性相反，故将两管栅极相连作为输入端，两个漏极相连作为输出端，则两管正好互为负载，处于互补工作状态。当输入低电平（Vi=Vss）时，PMOS 管导通，NMOS 管截止，输出高电平。当输入高电平（Vi=UDD）时，PMOS 管截止，NMOS 管导通，输出为低电平。两管如单刀双掷开关一样交替工作，构成反相器。与双极型逻辑电路相比，CMOS 逻辑电路具有以下优点：

（1）制造工艺简单，集成度和成品率较高，便于大规模集成；

（2）工作电源 UDD 允许变化的范围大，高、低电平分别为 UDD 和 0 V，抗干扰能力强；

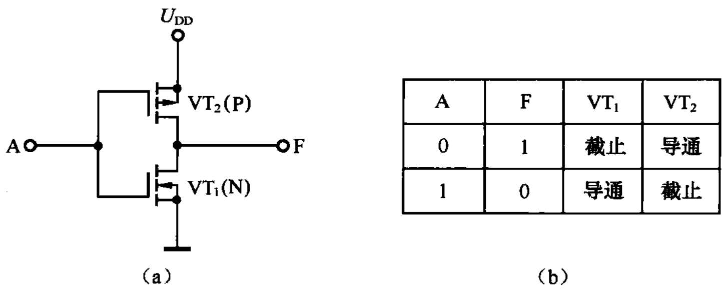
（3）在电源到地的回路中，总有 MOS 管截止，功耗特别低；

（4）输入阻抗高，一般高达 500 MΩ 以上，带负载能力强。

下面对 COMS 基本单元电路进行简单的介绍。

1. COMS 非门

图 5-1 所示为 CMOS 非门（即反相器）电路及工作状态。当 A 端电压为 0V 时，NMOS 管截止，PMOS 管导通，故 F 端电压等于 UDD。当 A 端电压为 UDD 时，情况正好相反，NMOS 管导通，PMOS 管截止，所以此时 F 端电压为 0V。

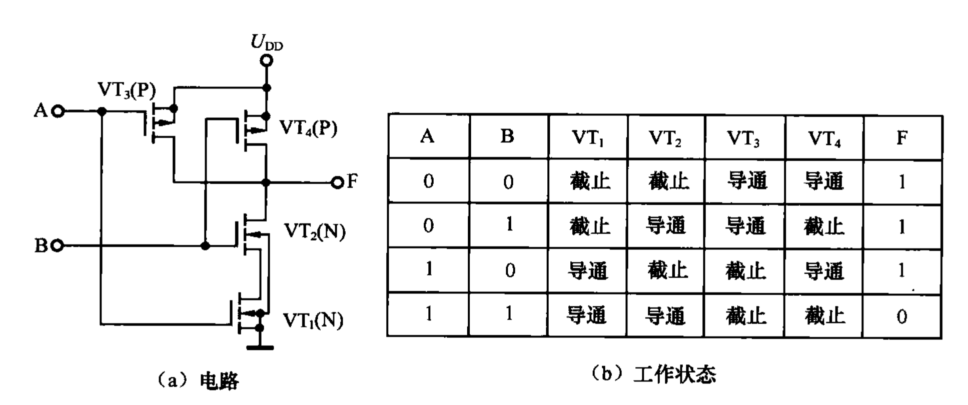


A	F	VT_1	VT_2
0	1	截止	导通
1	0	导通	截止

图 5-1　CMOS 非门电路及工作状态

2. COMS 与非门

CMOS 与非门电路及工作状态如图 5-2（a）所示。电路由 4 个 MOS 管组成，VT1 和 VT2 两个 NMOS 驱动管串联，VT3 和 VT4 两个 PMOS 负载管并联。当输入 A、B 至少有一个为低电平时，VT1、VT2 中就至少有一管截止，VT3、VT4 中就至少有一管导通，输出为高电平，F = 1；当输入 A、B 均为高电平时，VT1 和 VT2 都导通，VT3 和 VT4 都截止，输出为低电平，F = 0。所以，该电路实现了与非门的功能，输出 F 和输入 A、B 的逻辑关系如图 5-2（b）所示。



A	B	VT_1	VT_2	VT_3	VT_4	F
0	0	截止	截止	导通	导通	1
0	1	截止	导通	导通	截止	1
1	0	导通	截止	截止	导通	1
1	1	导通	导通	截止	截止	0

图 5-2　CMOS 与非门电路及工作状态

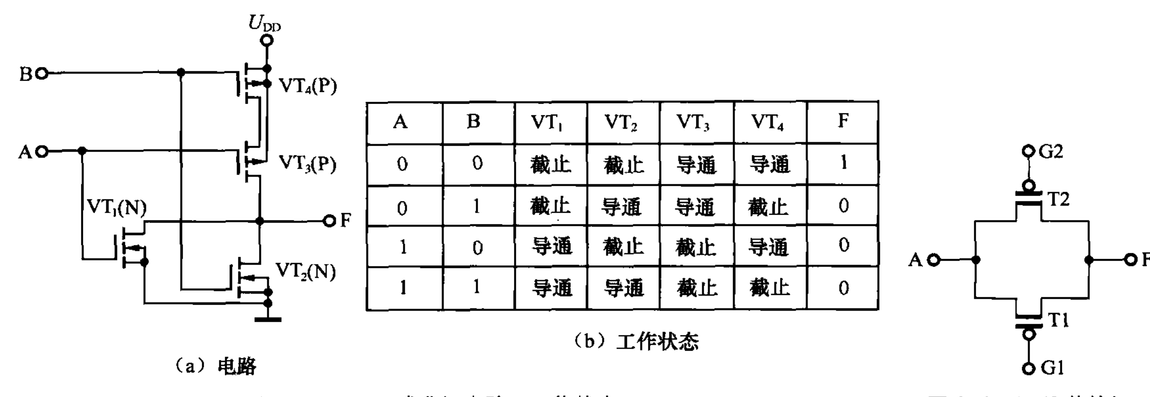
3. COMS 或非门

CMOS 或非门电路及工作状态如图 5-3（a）所示，其电路形式刚好和与非门相反，VT1 和 VT2 两个 NMOS 驱动管并联，VT3 和 VT4 两个 PMOS 负载管串联。当输入 A、B 均为低电平时，VT1 和 VT2 都截止，VT3 和 VT4 都导通，输出为高电平，因此 F=1；当输入 A、B 中至少有 1 个为高电平时，VT1、VT2 中至少有 1 个导通，VT3、VT4 中至少有 1 个截止，输出为低电平，因此 F= 0。可见，该电路实现了或非门的功能，输出 F 和输入 A、B 的逻辑关系如图 5-3（b）所示。

4. COMS 传输门

利用 MOS 管的源、漏极可以互换的特点，可以设计出一种用于实现 CMOS 电路级间耦

合的传输门，如图 5-4 所示。它由一个 NMOS 管和一个 PMOS 管并联而成，并联在一起的两端信号输入、输出端(可以互换)，两个管子的栅极分别由极性相反的 G1 和 G2 控制。当 G1=1（G2=0）时，传输门导通；而 G1=0（G2=1）时，传输门截止。CMOS 传输门的导通电阻和截止电阻相差甚大，当它驱动 MOS 负载时，接近于理想开关，可用来传输模拟信号，故又称为模拟开关。COMS 传输门的电路图如图 5-4 所示。



A	B	VT_1	VT_2	VT_3	VT_4	F
0	0	截止	截止	导通	导通	1
0	1	截止	导通	导通	截止	0
1	0	导通	截止	截止	导通	0
1	1	导通	导通	截止	截止	0

（b）工作状态

图 5-3　CMOS 或非门电路及工作状态　　　　图 5-4　COMS 传输门

根据 COMS 基本单元电路，可以设计出其他复杂的电路。这里举一个运用非门和传输门设计 D 触发器的例子。

D 触发器是最常用的时序器件，图 5-5 是由两个结构相同、触发极性相反的 D 锁存器构成的主-从 D 触发器。图中，传输门 1、2 和非门 5、6 构成了响应 CP 低电平的 D 锁存器，为“主触发器”；而传输门 3、4 和非门 7、8 构成了响应 CP 高电平的 D 锁存器，为“从触发器”。该电路的工作原理是，当 CP=0 时，传输门 1、4 导通，传输门 2、3 截止，输入信号 D1 可以进入“主触发器”，且“主触发器”的（内部）输出 Qm 为 D2，此时“从触发器”的输出锁定在 CP 变化前的 D2 值不变，而“从触发器”的输出 Q1=Qm1=D1，Q2=D2，这一状态一直要保持到下一次 CP 由 0 到 1 的时候，Q1 和 Q2 才会随 D1 变化。因此该电路是一个响应 CP 上升沿的 D 触发器。为保证该电路正常工作，输入信号 D 与 CP 间需要满足一定的建立时间，但无需保持时间。

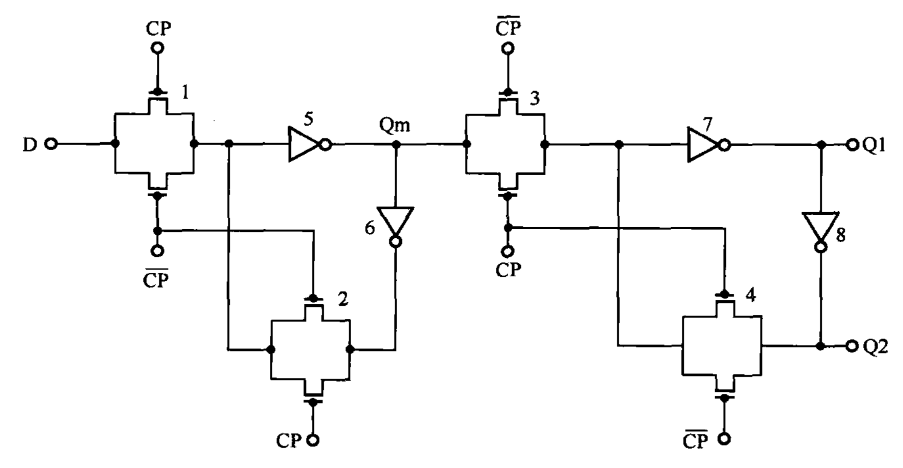


图 5-5　D 触发器

5.3 专用集成电路设计

专用集成电路（ASIC）是一种为专门目的而设计的集成电路，自20世纪80年代中期以来，ASIC得到广泛的应用和重视，ASIC的蓬勃发展正推动着设计方法学和设计工具的完善，同时促进了系统设计人员和芯片设计人员的结合和相互渗透。

5.3.1 集成电路的设计路线

IC有两种设计路线：“自底向上”（Bottom-up）和“自顶向下”（Top-down）。

1. “自底向上”（Bottom-up）的设计路线

“自底向上”的设计路线应该说是整个IC发展的基本路线。从根本上来说，IC研究是从最基础的半导体理论、材料开发、工艺研究与实验、新器件的发明、电路设计到系统应用这样一种“自底向上”的路线进行的。IC设计，特别是在IC发展前期，所遵循的也是“自底向上”的路线设计，即从工艺开始，先进行单元设计，在精心设计好各单元后逐步进行功能块、子系统设计直至最终完成整个系统设计。事实上，在IC发展前期，集成电路制造只能达到功能单元的中小规模水平，所以在IC制造厂家，不论是模拟还是数字电路，其设计基本上都是自最底层的工艺出发，向上直到IC功能模块的实现。系统厂家则利用IC模块完成系统设计。到目前为止，在模拟IC和较简单的数字IC设计中，大多仍然采用“自底向上”的设计方法。

随着IC技术的发展，IC功能模块种类的不断增加和性能的不断完善，IC设计的重点向更大规模、更复杂性能的系统转移。特别是数字IC，不管其逻辑和电路多么复杂，都可以由品种不多的功能模块和门电路构成。从逻辑功能、电路网表到版图都形成了完备的单元库，这些功能模块和门电路早已开发出来。可以说，IC设计底层的工作已经完成，加上EDA工具的开发，使电路设计甚至是系统设计完全有可能从顶层，即更复杂电路和系统出发，“自顶向下”地完成IC的设计。

2. “自顶向下”的设计

“自顶向下”的设计有以下几种：

（1）行为设计。

首先设计者需要进行行为设计，以确定芯片的性能、拟采用的工艺以及允许的芯片面积和成本等；其次进行结构设计，根据芯片的特点，将其分解为接口清晰、相互关系明确的子系统，这些子系统可能包括模拟单元和数字系统。

（2）逻辑图、电路图设计。

把各子单元转换成逻辑图或电路图。对模拟单元直接进行电路设计；对数字系统则先进行逻辑设计，当逻辑设计经验认证为正确后再将其进一步转换成电路图。无论是模拟电路还是数字系统，电路设计阶段均与设计选用的工艺紧密相关。设计者应该根据制造厂家提供的工艺参数，选择合适的器件模型和模拟工具，以确定电路图是否满足设计要求。

（3）版图设计。

版图设计，即将电路图转换成版图。与电路设计一样，版图设计也是同工艺密不可分的。

设计者必须按照来自制造厂家的几何设计规则进行电路图的版图设计。

受到 IC 制造工艺极限条件和具体工艺要求的限制，IC 版图设计在移交制造厂家前必须进行一系列的版图验证，以确保芯片的成品率。版图数据主要验证过程如下：首先对版图进行几何设计规则验证（Design Rule Check，DRC）；DRC 检查无误后，再对版图做电气规则验证（Electrical Rule Check，ERC）；最后进行版图与电路图一致性验证（Layout Versus Schematic，LVS）。

当集成电路工艺发展到深亚微米水平后，版图中连线的寄生效应对电路性能的影响变得不容忽视。因此版图后模拟（Post Simulation）成为 DRC、ERC、LVS 版图验证后必不可少的一个环节。为了进行后模拟（后仿真），首先要根据工艺参数从设计的版图提取寄生电容、电阻值等寄生参数，这就是寄生参数提取过程（Layout Parameter Extraction，LPE）。将提取的寄生参数回代到原电路设计中再模拟（即后模拟），并适当调整原电路参数以得到满意的模拟结果。根据调整后的电路参数修改相应版图后，重复 DRC、ERC、LVS 验证，LPE 及后模拟。一个设计往往需要多次反复这个过程才能达到满意的结果。只有在所有的检查都通过并被证明正确无误后，才能将版图结果转换为标准的版图数据文件，送交制造厂家。

这里需要指出，以上关于“自底向上”（Bottom-up）和“自顶向下”（Top-down）两种设计路线的讨论只是从方法的角度来进行的。实际上，一个芯片的设计往往需要把“自顶向下”和“自底向上”两者结合起来共同完成。

从另一个角度出发，一个完整的 IC 设计过程可以认为是由行为域、结构域和物理域 3 个设计域构成。行为域描述一个特定的系统做些什么，要完成什么功能；结构域描述实现某一功能的具体结构以及各组成部件是怎样连接在一起的；物理域描述结构的物理实现，即怎样实现制造一个满足一定的连接关系的结构并能实现所要求功能的芯片。图 5-6 和图 5-7 分别给出了 VLSI 数字 IC 和模拟 IC 的典型设计过程框图。图 5-6 采取的是“自顶向下”的设计路线；而图 5-7 则包含了“自顶向下”和“自底向上”的路线。另外，两个设计图中均包括了从后道设计步骤进行修改的循环过程。

通常，人们把集成电路设计方法分为全定制设计方法与半定制设计法两大类。而半定设计法中又分为 5 种不同的方法，集成电路设计人员可以根据不同的要求选择各种设计方法。下面分别对各种设计方法做简要介绍。

5.3.2　全定制设计方法

全定制设计方法（Full-Custom Design Approach）使用于要求得到最高速度、最低功耗和最省面积的芯片设计中。全定制设计方法有时也称为全用户设计方法和用户设计方法。这种设计方法完全是由用户设计师根据所选定的生产工艺按自己的要求独立地进行集成电路产品设计，这样可以使所设计的电路具有尽可能高的工作速度、尽可能小的芯片面积和成本。设计人员要对体系结构、逻辑、电路等各个层次进行精心的设计，对不同方案进行反复比较，特别要对影响性能的关键路径做深入的分析。一旦确定以后就进入全定制版图设计阶段，全定制版图设计的特点是针对每个晶体管进行电路参数和版图优化，以获得最佳的性能（包括速度和功耗）以及最小芯片面积。由于这种设计方法的版图布局和布线都要用人工布置，并要求尽可能紧凑和改正设计错误也是非常艰巨的工作。

随着硅工艺技术的发展和设计自动化程度的不断提高，EDA 工具提供了大量的、经过精

心设计好的标准化单元，使得全定制设计方法得到越来越广泛的应用。目前CMOS模拟集成电路所采用的设计方法，基本上属于全定制方法。

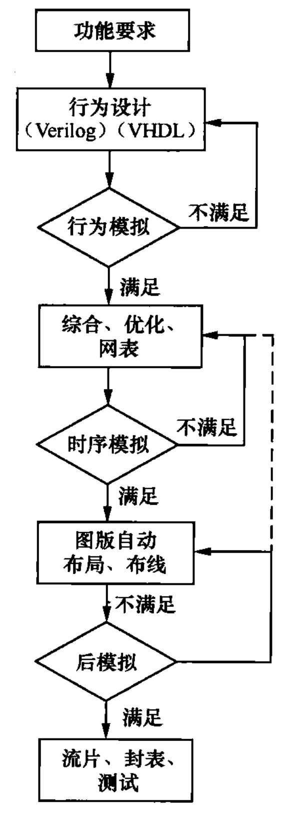


图5-6 VLSI数字IC的设计流程图

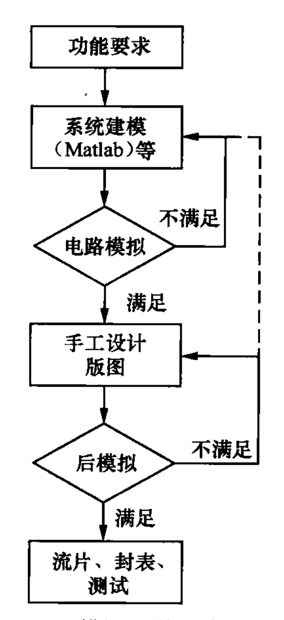


图5-7 模拟IC的设计流程图

5.3.3 半定制设计方法

半定制设计方法（Semi-Custom Design Approach）可以分为门阵列（Gate Array，GA）法、门海（Sea Of Gate，SOG）法、标准单元（Standard Cell，SC）法、积木块（Building Block，BB）法、可编程逻辑器件（Programmable Logic Device，PLD）法。

1．门阵列设计法

门阵列是指在一个芯片上把形状和尺寸完全相同的单元排列成阵列，每个单元内部含若干器件，单元之间留有纵向尺寸固定的布线通道。通过连接单元内的器件使得每个单元实现某类门的功能，再通过各单元之间的连接实现电路的设计要求。门阵列的基片四周有固定数目的输入/输出单元和压焊块。

2．门海设计法

门海设计技术是把一对不共栅的P管和N管组成的基本单元铺满整个芯片(除I/O区外)。基本单元之间无氧化隔离区，布线通道不确定，可将基本单元链改成无用器件走线，宏单元连接线在无用器件区上进行。与门阵列法相比，门海法具有门利用率更高、集成密度更大、

布线灵活和保证布线布通率更好等优点。但它仍然有不足之处：一是它仍有布线通道，而且增加的布线通道只能是基本单元高度内所含通道数的整数倍，这往往使增加的通道数超过实际的需要，造成面积浪费；二是布线通道下的晶体管不能再用来实现逻辑，因此门的利用率仍不是很高。

3．标准单元设计法

标准单元设计是一种库单元设计方法。这种方法的特点是各个单元具有同一高度（指版图尺寸），但宽度不等。单元本身经过精心设计，并完成了设计规则检查和电学性能验证。设计好的各单元存入设计系统的物理单元库中以便调用，单元的逻辑符号及电学特性则存入逻辑库中。利用标准单元进行设计时，设计者将所需要的单元从标准单元库中调出来，并排列成行，行间留有可调整的布线通道，再按设计电路的功能要求将各内部单元以及输入/输出单元连接起来，就得到所需的芯片版图。

4．积木设计方法

积木块设计方法（BBL-Building Block Layout）是另一种库单元设计方法，又称通用单元设计法。与标准单元法不同之处是：

（1）积木块设计方法既不要求每个单元（或称积木块）等高，也不要求等宽。每个单元可根据最合理的情况单独进行版图设计，从而可获得最佳性能。设计好的单元存入库中备用。

（2）积木块设计方法没有统一的布线通道，而是根据需要加以分配。

5．可编程逻辑器件的设计方法

可编程逻辑器件的设计方法是指用户通过对“与”、“或”矩阵进行掩膜编程，得到所需的专用集成电路。可编程逻辑器件 PLD 包含两个基本部分：一是逻辑阵列，另一个是输出单元宏单元（Macrocell）。逻辑阵列式用户可编程的部分，它由“与”矩阵、“或”矩阵和反相器组成。宏单元可以让设计者改变 PLD 的输出结构。

5.4 专用集成电路设计的 EDA 技术

现代电子设计的核心是 EDA 技术。EDA 技术就是依靠功能强大的电子计算机，在 EDA 工具软件平台上，用硬件描述语言（HDL）作为系统逻辑描述手段完成的设计文件，由工具软件自动地完成逻辑编译、化简、分割、综合、优化、仿真直至下载到可编程逻辑器件 CPLD/FPGA 或专用集成电路 ASIC 芯片中，实现既定的电子电路设计功能。EDA 技术使得电子电路设计者的工作仅限于利用硬件描述语言和 EDA 软件平台来完成对系统硬件功能的实现，极大地提高了设计效率，缩短了设计周期，节省了设计成本。

ASIC 的 EDA 工具分为三大类：一是设计输入和数据管理工具，帮助设计者输入设计对象、设计要求，管理设计数据，如 HDL 编辑与编译、逻辑图编辑、版图编辑、数据库管理等；二是模拟验证工具，帮助设计者验证设计的正确性，包括系统模拟、逻辑模拟、电路模拟、时序分析、设计规则检查、电学规则检查、版图与电路一致性检查、版图参数提取等；三是综合设计工具，帮助设计者完成各层次的设计，如系统综合、行为综合、逻辑综合、版

图自动布局布线等。

5.4.1　输入的设计

设计者对待设计系统的功能描述、设计构思，直至设计结果均需通过 EDA 输入工具录入计算机，从而加载到后续的各种 EDA 工具上。

输入方式可以使原理图和 HDL 文本，且 HDL 更为常用，因为它既便于高层次的仿真验证，又便于综合、优化。有些工具以 HDL 为基础，同时还支持一些非编程的图形化输入方式，如框图、流程图、数据流图、状态图、真值表、波形图等，这些输入方式经编译后最终都将转化成 HDL 文本，供后续处理。

版图编辑器是全定制版图设计所需的交互式输入工具。它可直接作为物理版图的输入工具，也可作为版图综合和自动布图后的显示与修改用的辅助工具。进行版图设计时，往往采用“自下而上”的层次式设计方法，先设计出整个芯片或待设计部分的底层部件的版图，在设计过程中可能用到已有的库单元或标准单元，也可能需要进行交互式设计，然后以层部件为基础进行高一级的设计，直至完成整个芯片版图的设计。版图编辑器的主要功能为图形编辑，此外还具有与版图设计有关的配置、层次化操作、库管理、交互式设计规则检查等。

数据文件也是一种常用的输入方式，除用于同一设计环境中作为中间数据在不同软件包之间交换信息外，还可以在不同设计工具之间建立联系，以便设计人员利用不同厂家的设计工具进行设计。为便于各设计工具间的数据交流与共享，专门定制了一些标准数据交换格式，如用于电路网络表描述的 EDIF（兼有版图描述功能）、Spice、CDL 格式和用于版图描述的 CIF、GDSII 格式等。

5.4.2　设计验证

在集成电路的设计过程中，需要完成两方面的任务：一是根据系统的功能要求经设计综合得出相应的电路结构：二是对得到的电路进行检查，以验证所设计的电路确实满足了指标要求。验证的方法有 3 种：模拟（仿真）、规则检查和形式验证。规则检查是分析电路设计结果中各种数据的关系是否符合设计规则。形式验证是近几年来兴起的一种验证方法，它利用理论证明的方法来验证设计结果的正确性，目前这种方法尚未成熟。

所谓模拟，是指从电路描述抽象出模型，然后将外部激励信号或数据施加于此模型，通过观察该模型的响应来判断该电路是否实现了预期的功能，模拟方法是目前最常用的验证方法。

系统的综合或设计过程分若干层次进行，对于每个设计层次，都有相应的验证工具对设计结果进行检查，按设计层次的不同可将设计验证分为高层次仿真（系统或行为级仿真）、逻辑模拟与时序模拟、定时分析、电路模拟和版图验证。

1．高层次仿真

自从 HDL 出现后，特别是 1987 年 VHDL 成为 IEEE 标准以来，高层次模拟得到了迅速发展，为系统级和行为级设计验证提供了很大的便利。如在系统结构设计阶段，采用行为级仿真工具，可使设计者方便地进行系统结构的比较和设计方案的优化，因为在行为级设计中，功能修改只需改写若干条行为描述语句。

2. 逻辑模拟与时序模拟

逻辑模拟，按模拟时器件的规模和类型的不同，又可以分为功能块级、逻辑门级、开关级3种。功能级模拟把寄存器、存储器、输入/输出控制器件、总线以及组合逻辑运算器件等当作基本电路单元，用来检查数据在各寄存器中的传输情况。门级模拟把各种逻辑门（触发器、计数器及译码器等）当作基本电路单元，用来检查逻辑设计的正确性。开关级仿真把每个晶体管都当作一个独立的开关，用来模拟硬件中信号强度对逻辑设计的影响。通常开关级仿真比逻辑门级以及功能级仿真的精度高，但其计算成本也高。

功能逻辑模拟不考虑信号通过逻辑模块的延迟时间。因此，它只能验证系统的逻辑行为是否正确，而没有考虑实际应用中非常重要的时序问题。

电路设计中，除要满足逻辑功能正确性外，还应保证各信号时序的正确性。时序模拟是专用时序分析的工具。通常的时序模拟工具能够检查电路中的竞争、险象及振荡现象。有些时序分析工具还可以做最坏情况分析。如通过对寄存器的数据通路取最大延迟、对时钟路径取最小延迟，便可检查出最坏情况下该寄存器的建立时间能否满足。

3. 定时分析

定时分析不同于逻辑模拟与时序模拟。定时分析只考虑所有可能的信号路径的延迟，而逻辑模拟与时序模拟是以特定的输入信号来控制模拟过程的，因而只能检查特定输入信号的传输路径延迟。对于多数数学电路来说很难保证通过某些给定的输入信号就能够对电路的各条路径进行完备的测试。定时分析工具采用跟踪信号路径法取代由特定输入信号模拟电路的方法来测试所有可能的路径的信号延迟。定时分析工具还具有关键路径分析功能，以检查时序单元输入数据端可能存在建立时间与保持时间的错误。

4. 电路模拟

数字电路本质上仍是模拟电路（由晶体管、电阻、电容等模拟元器件构成）。在逻辑设计完成之后，需要进行电路设计（若逻辑设计中采用了单元库中的单元，则其电路、版图均可从库中提取）。电路设计的任务是根据所要求的电路性能，如速度、功耗、电源电压、逻辑操作类型、信号电平的容限等确定电路的结构和各元器件的参数。同时应考虑工艺上可能发生的偏差和使用温度上的变化等因素，使所设计的电路仍能达到规定的性能。目前，还难以借助 EDA 工具进行全自动的电路设计，实际做法往往是设计者根据电路框图，进行电路结构的设计并初步确定元器件参数，然后对该电路进行计算机模拟分析，再根据分析结构进行修改，经多次反复，最后得到符合要求的电路。

5. 版图验证

版图设计完毕后，为尽可能保证设计的正确性，避免因设计错误造成巨大的经济损失，在交付厂家生产前，需对版图进行彻底的检查，这一过程称为版图验证。它包括设计规则检查 DRC（Design Rule Check）、网络表及参数提取 NPE（Netlist&Parameter Extraction）、电学规则检查 ERC（Electric Rule Check）、版图与电路一致检查 LVS（Layout Versus Schematics）以及后模拟。由于版图层面多、图形繁杂，因而上述验证工作主要依靠 EDA 工具完成。

5.4.3　设计综合

数字系统设计实际上是不同层次，不同描述形式的转化。“自上而下”的设计过程则是将系统级的行为描述转化为逻辑级结构描述（逻辑设计）或版图级几何描述（ASIC 设计）。采用“综合”这个词是指靠计算机自动完成的设计，以便与人工进行的“设计”相区别，最早出现的综合是逻辑综合与版图综合，目前已较为成熟。在这两个设计环节，除了有特殊要求的电路设计外，一般均采用自动综合加以实现，系统级综合与算法综合（又称为行为综合）称为高层次综合（或高级综合），目前尚不成熟，还有待进一步完善。这两个设计环节目前仍以人工设计为主。当然也可借助高层次综合工具产生多种设计方案，从中选出最佳方案进行后续设计。

除了上述几种综合工具外，还有可测性综合、功耗综合、工程修改（Engineering Change Orders，简称 ECO）综合以及深亚微米综合等。

1．高层次综合

目前，高层次综合主要是指从算法级行为描述到实现它的寄存器传输级结构描述的转换。该 RTL 结构包括一个内含寄存器、运算器等功能电路构成的数据处理单元和一个控制数据处理单元工作时序的控制器。通常有多种给定行为的硬件结构，高层次综合的一个重要任务就是找出一个满足约束条件和目标集合且开销最小的硬件结构。

2．逻辑综合

选定 RTL 结构后，逻辑综合完成硬件的数据流图向门级结构描述的转换。逻辑综合的主要任务是在一个包含众多结构、功能、性能均已知的逻辑元件的逻辑单元库的支持下，寻找出逻辑功能电路的最佳（至少是较佳）实现方案。它包括两方面的内容：一是逻辑结构的生成与优化，主要是进行逻辑化简与优化，以便在满足逻辑功能的前提下，用尽量少的元件和连线形成网络结构（逻辑图）；二是逻辑网络的性能优化，利用给定的单元库，对已生成的逻辑网络进行元件配置，进而估算电路的工作速度、功耗及所占面积。速度与面积或速度与功耗是矛盾的。这一步允许设计师对它们作出折中，以确定合适的元件配置，得出最终的、符合设计要求的逻辑网络结构。

3．版图综合

版图综合是指门级的逻辑结构描述向物理版图描述的转换。由于单元电路的版图目前尚不能自动综合生成，因此版图综合实际上是布图综合，即对各单位电路自动进行布局与布线并完成相互间的布线。而单元电路的版图要么是设计库中已有的，要么需人工设计。版图综合按照门级、速度和功耗 3 个指标进行布局与布线的优化。版图综合不能与逻辑综合独立，否则难以使约束条件与目标集合全部达到要求。一般需在两者间建立自动交互的设计环境，即用布图的拓扑信息和参数返标到相应的设计环节作为逻辑综合的约束条件。这样，一方面能把逻辑综合最佳地连接到版图综合，另一方面由于实际版图及参数的返标将加快设计过程的收敛。

4．测试综合

随着芯片系统复杂度的日益提高，可测性设计显得越来越重要，测试综合包括可测试结

构的嵌入与自动测试图形生成。可测性设计需在设计前期就加以考虑。在设计综合阶段所生成的电路应包括扫描测试的附加电路。当整个电路设计完成时，可测试设计也随之完成，这样便可在保证性能与面积要求的前提下，实现可测试电路设计。

测试综合往往与逻辑综合集成在一起使用。这样在测试综合中自动生成测试图形时，产生高覆盖率的测试代码，并为电路与系统的设计提出一种无冲突的自动测试方案。测试综合提供了这种经过验证又可以对测试结果进行预测的有效手段。它可以使设计师不必因顾及系统的可测试性而重新设计，因而在缩短设计周期的同时还可以大大减少未来用于测试方面的费用。

5.5 设计实例分析

本节以可编程分频器为例，介绍专用集成电路的后端设计方法。

5.5.1 可编程分频器原理

可编程分频器的输入信号为逻辑控制信号 CtrlLogic 和 DMP 的输出信号 ClkFront，两个预置值 N_1 和 N_2 可产生供 PFD 鉴频鉴相的 4MHz 的数字脉冲 LowOut，输出信号 ModCtr 用来控制 DMP。

可编程分频器的逻辑控制由 5-bit 的 N_1 和 4-bit 的 N_2 两个预置值实现，置入预置值后，可编程分频器在 ClkFront 的控制下计数，当计数值小于 N_2 时，可编程分频器通过 ModCtr 信号控制 DMP 进行 9 分频，当计数值在 N_2 到 N_1 之间时，可编程分频器控制 DMP 进行 8 分频，其分频原理如图 5-8 所示。最终的分频比可表示为 $M=N_2\times9+(N_1-N_2)\times8=N_1\times8+N_2$，改变 N_1 和 N_2 值就可以实现不同的分频比。本设计中 N_1 取值为 32，N_2 取值范围为 3～10，通过改变 N_2 就可以和 DMP 一起实现 259～266 的 8 个不同的分频比，完成对 4 分频器的分频，最终使 VCO 输出间隔为 4MHz 的 4.144～4.256GHz 的 8 个不同频点的振荡信号。

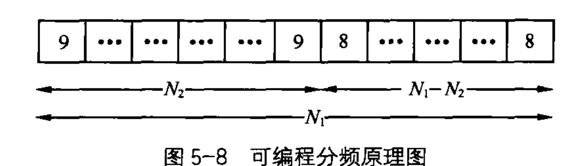


图 5-8 可编程分频原理图

5.5.2 可编程分频器的后端设计

在进行设计之前先介绍一下将要用到的软件工具。

（1）VCS：VCS 是编译型 Verilog 模拟器，它完全支持 OVI 标准的 Verilog HDL 语言、PLI 和 SDF。VCS 具有目前行业中最高的模拟性能，其出色的内存管理能力足以支持千万门级的 ASIC 设计，而其模拟精度也完全满足深亚微米 ASIC Sign-Off 的要求。

（2）Apollo II：Apollo II 是世界领先的 VDSM 布局布线工具。它能对芯片集成系统的 VDSM 设计进行时序、面积、噪声和功耗的优化。Apollo-II 的优点如下：

① 使用专利布局布线算法，产生出最高密度的设计；

② 使用先进的全路径时序驱动的布局布线、综合时钟树算法和通用时序引擎，获得快速时序收敛；

③ 与 Saturn 和 Mars 一起使用，可提供对时序、功耗和噪声的进一步优化；

④ 应用了如天线和连接孔等先进特性，能适应 VDSM 的工艺要求；

⑤ 高效强大的ECO管理和递增式处理，确保最新的设计更改能快速实现。

（3）Cadence：Cadence Allegro系统互连平台能够跨集成电路、封装和PCB协同设计高性能互连。应用平台的协同设计方法，工程师可以迅速优化I/O缓冲器之间和跨集成电路、封装和PCB的系统互联。该方法能避免硬件返工并降低硬件成本和缩短设计周期。约束驱动的Allegro流程包括高级功能用于设计捕捉、信号完整性和物理实现。由于它还得到Cadence Encounter与Virtuoso平台的支持，Allegro协同设计方法使得高效的设计链协同成为现实。

（4）Hspice：Hspice是Meta-Software公司为集成电路设计中的稳态分析、瞬态分析和频域分析等电路性能的模拟分析而开发的一个商业化通用电路模拟程序，它在柏克莱公司的SPICE（1972年推出）、MicroSim公司的PSPICE（1984年推出）以及其他电路分析软件的基础上，又加入了一些新的功能，经过不断的改进，目前已被许多公司、大学和研究开发机构应用。Hspice可与许多主要的EDA设计工具，如Candence，Workview等兼容，能提供许多重要的针对集成电路性能的电路仿真和设计结果。

采用Hspice软件可以在直流到高于100MHz的微波频率范围内对电路作精确的仿真、分析和优化。在实际应用中，Hspice能提供关键性的电路模拟和设计方案，并且应用Hspice进行电路模拟时，其电路规模仅取决于用户计算机的实际存储器容量。

对于以上软件的介绍比较简单，有兴趣的读者可以翻阅相关的资料，这里就不多做介绍了。

本设计首先由HDL代码得到行为描述的网表，再由DC实现初步综合，根据设计要求对综合主要作了时钟和输入输出的约束，约束值如表5-1所示。因为电路实际工作周期为8ns，考虑到后面的布线及裕量要求，在综合时将时钟周期约束设为4ns；对于时钟网络，用set_clock_latency和set_clock_uncertainty将时钟延迟、时钟抖动和偏斜分别设置为1ns和0.01ns。为了保证时钟网络在综合时不被修改，将时钟ClkFront设置为不需改变的网络，同时将max area设置为0以实现面积的最小化。

表5-1 综合的约束设定

约束内容	DC中的相应属性	设置值
时钟周期	clock period	4ns
时钟延迟	clock latency	1ns
时钟抖动和偏斜	clock uncertainty	0.01ns
不需改变的网络	dont touch network	ClkFront
驱动无限大的网络	net drive	ClkFront
面积约束	max area	0
输入延时	input delay	0.8ns
输出延时	output delay	0.8ns

由于设计规模不大，得到优化的门级网表后，直接将延迟信息SDF返标到验证工具VCS中进行综合后的功能仿真和验证。

得到正确的验证结果后，布局布线是在Apollo II中完成。在进行版图规划时，先要设定芯片的面积和宽长比等。该电路由于输入输出较多，芯片面积主要由引脚数量决定。芯片内核中除了放置标准单元外，还要留下标准单元行间的空隙、布线通道、时钟树缓冲单元放置空间、电源环空间等额外的面积。

确定P/G管脚数时，可根据经验每6～8个输入输出设置1～2对P/G引脚，也可通过计算得到I/O引脚的个数。先查得单元库中I/O引脚可承载的最大电流Icore_power_pad=26mA，则I/O电源引脚个数$N_{core_pad}=I_{core}/I_{core_power_pad}$=11.1/26=1对，本设计中考虑到在片测试时探针的要求，将P/G引脚设置成1对。

接下来设定电源环线的宽度。根据DC输出的功率报告可知内核功耗P_{core}=12.25mW，若考虑50%的裕量要求，则符合设计要求的内核功耗应为20mW。TSMC0.18μm CMOS工艺的内核电压V_{core}为1.8V，则$I_{core}=P_{core}/V_{core}$=20/1.8=11.1mA。

考虑到金属的电迁移效应，由于内核为正方形且有2对供电引脚，则有$I_{strap}=I_{core}/2$=5.6mA。由工艺文件查得电流密度J_{metal}=1mA/μm，方块电阻值R_{m6}=0.036Ω/sq，从而得到电源环的最小线宽$W_{strap}=I_{strap}/J_{strap}$=5.6μm，则可得IR压降$V_{drop_strap}=I_{strap}\times R_{strap}=R_{m6}\times J_{metal}\times L_{strap}$=0.022V<0.09V，满足5%$V_{DD}$的要求。

完成布图规划后，即可进行时序约束下的子单元布局，这样，子单元的实际布局受到了时序的约束，以达到设计不违犯时序要求。各个子单元的电源和地的引脚通过电源带线连到电源环上。

布局之后的工作是时钟树综合，这里用到了由前端综合时生成的时钟约束条件。

经过全局布线和详细布线，还要对电路作静态时序分析。具体做法是将Apollo中提取的set_load文件、SDF文件和含有时钟信息的DSPF文件加载到Synopsys的PT中进行静态时序分析，分析结果表明，保持时间（hold time）和建立时间（setup time）均满足设计时序要求。

当确定不存在时序问题后，即可将Apollo的布局布线结果导出并转换为Cadence格式，接下来进行的DRC和LVS都是在Cadence设计环境中完成的。

除了功能验证之外，进一步提取RC参数并用Hspice做晶体管级的瞬态分析，图5-9所示为电路的瞬态分析波形。

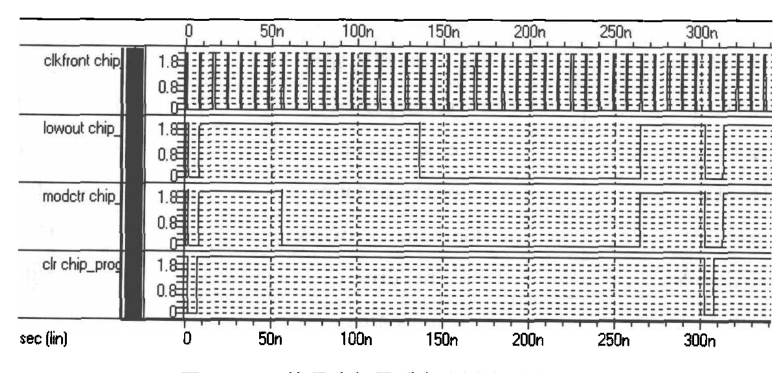


图5-9　可编程分频器瞬态分析波形（N=6）

5.5.3　芯片验证与测试

设计的可编程分频器可用TSMC 0.18μm CMOS工艺实现并可流片，图5-10所示为其显微照片，芯片面积为0.66mm，内核面积1360.5μm。

测试时将芯片输出信号接到示波器50Ω档进行测试。芯片的功耗测试结果表明，测试电流6.8mA和仿真结果7.5mA基本匹配，测试功耗13.4mW与仿真结果12.25mW相当接近。

分频精度测试时，示波器读出的控制信号 ModCtr 和分频输出信号 LowOut 在各个分频点上均与理论值相当吻合，图 5-11 所示为 N_2=8 时的测试波形，控制信号 ModCtr 的占空比为 25.13%，分频输出信号 LowOut 的占空比为 50.96%，正确实现了分频功能。

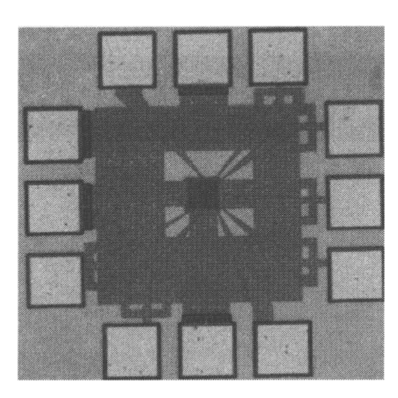


图 5-10　可编程分频器的显微照片

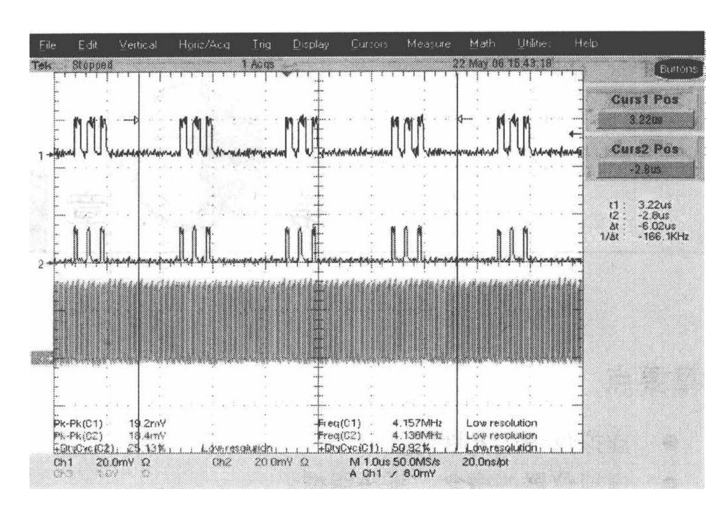


图 5-11　可编程分频器测试波形输出（N_2=8）

习　题　五

1. 简述集成电路设计的分类。
2. MOS 工艺的有哪几种常用工艺？
3. 试设计基本 RS 触发器的 CMOS 电路。
4. 简述全定制集成电路的设计过程。
5. 设计仿真与综合分成哪些层次？具体内容是什么？
6. 版图验证包括哪些内容？
7. 简述 EDA 技术在 ASIC 中的应用。

第6章 虚拟仪器系统及设计

本章要点

- 虚拟仪器的发展现状
- 虚拟仪器的概念、系统组成
- LabVIEW 的特点及创建虚拟仪器过程
- 基于 LabVIEW 的虚拟仪器的应用设计举例

6.1 虚拟仪器的发展状况

本节介绍了虚拟仪器的研究背景及国内外在虚拟仪器方面的研究现状，描述了虚拟仪器的诞生，发展历史及近年来的发展情况。

6.1.1 虚拟仪器在国外的发展状况

虚拟仪器利用个人计算机强大的图形环境和在线帮助功能，建立虚拟仪器面板，完成对仪器的控制，数据分析与显示，用户可以根据自己的需要定义仪器的功能。虚拟仪器概念最早是由美国国家仪器公司（NI）在 1986 年提出的，但其雏形可以追溯到 1981 年由美国西北仪器系统公司推出的 Apple II 为基础的数字存储示波器，由于当时计算机软件开发水平的限制，编写个人仪器的驱动程序和人机交互接口是一项专门的技术工作，必须由专业厂商才能完成，这种状况使得个人仪器的推广和应用没有形成工业标准。从 20 世纪 80 年代中期开始，随着微软公司 Windows 操作系统的出现，使得计算机操作系统的图形支持功能得到很大提高。1986 年，美国国家仪器公司推出了图形化的虚拟仪器编程环境 LabVIEW，标志着虚拟仪器设计软件平台基本成型。国际上从 1988 年陆续有虚拟仪器产品面市，当时有五家制造商推出 30 种产品。此后，虚拟仪器产品每年成倍增加，到 1994 年底，虚拟仪器制造厂已达 95 家，共生产 1000 多种虚拟仪器产品，销售额达 2.93 亿美元，占整个仪器销售额 73 亿的 4%。美国是虚拟仪器的诞生地，也是全球最大的虚拟仪器制造国，生产虚拟仪器的主要厂家有 HP 公司，目前生产 100 多种型号的虚拟仪器，Tektronix 公司目前生产约 80 多种型号的虚拟仪器，此外还有 NI 公司、Keithely 公司等。目前虚拟仪器在发达国家已经十分普及，在美国虚拟仪器系统及其图形编程语言，已作为各个大学理工科学生的一门必修课程。

6.1.2 虚拟仪器在国内的发展状况

LabVIEW 作为虚拟仪器开发系统的杰出代表，在我国虽然引进的时间不长，但是现在已经被科技工作者认识、推广和应用，促进了中国测试领域的技术革命。它被许多企业、科研单位用于产品测试和测控系统。另外，包括一些著名高校在内的许多学校不仅建立了基于虚拟仪器的实验室，而且还开设了 LabVIEW 编程的课程。20 世纪 90 年代中期以来，中国国防科技大学、清华大学、中科泛华测控技术有限公司等很多院校和高科技公司，在研究和开发虚拟仪器设计平台方面做了一系列有益工作，取得了巨大成效。例如，清华大学汽车系利用虚拟仪器技术构建的汽车发动机检测系统，用于汽车发动机的出厂检验，主要检测发动机的功率特性、负荷特性等；华中理工大学机械学院工程测试实验室将其虚拟实验室成果在网上公开展示，供远程教育使用；四川联合大学基于虚拟仪器的设计思路，研制了“航空电台二线综合测试仪”，将 8 台仪器集成于一体，组成虚拟仪器系统；复旦大学、上海交通大学、广州暨南大学等一批高校，也开发了一批新的虚拟仪器系统并用于教学和科研，得到了很好的应用。国内专家预测：未来的几年内，我国将有 50%的仪器为虚拟仪器。国内将有大批企业使用虚拟仪器系统对生产设备的运行状况进行实时检测。随着微型计算机的发展，虚拟仪器将会逐步取代传统的测试仪器而成为测试仪器的主流。虚拟仪器技术的提出与发展，标志着二十一世纪自动测试与电子测量仪器领域技术发展的一个重要方向。

21 世纪初，世界虚拟仪器的生产厂家已超过千家，其品种将达到数千种，市场占有率越来越高。虚拟仪器将成为本世纪仪器发展的方向，而且有逐步取代传统硬件化电子仪器的趋势。

6.2 虚拟仪器技术简介

本节主要介绍了虚拟仪器的基本概念、系统组成以及它和传统仪器相比所具有的优点。通过本节的学习，读者可以明白什么是虚拟仪器，虚拟仪器的构成以及它的特点。

6.2.1 虚拟仪器概念

随着电子技术的发展，电子测量仪器经历了由模拟仪器、分立组件式仪器、数字化仪器、带 GPIB 接口的智能化仪器到全部可编程虚拟仪器的发展历程。近些年来，随计算机科学和微电子技术以及网络技术的迅速发展和普及，有力地推动了多年来发展相对缓慢的仪器技术的更新和进步。与此同时，仪器的远程控制、实验信息的远程获取和传输也越来越重要，于是一种新型的，基于计算机技术所形成的仪器种类—虚拟仪器（Virtual Instrument，VI）技术出现了。它不仅被广泛地应用在科学研究领域，而且使得新型远程教育模式的实现成为可能。

虚拟仪器技术就是利用高性能的模块化硬件，结合高效灵活的软件来完成各种测试、测量和自动化的应用。自 1986 年问世以来，世界各国的工程师和科学家都开始将 NI 的 LabVIEW 图形化开发工具用于产品设计周期的各个环节，从而改善了产品质量，缩短了产品投放市场的时间，并提高了产品的开发和生产效率。使用集成化的虚拟仪器环境与现实世界的信号相连，分析数据以获取实用信息，共享信息成果，有助于在较大范围内提高生产效率。虚拟仪器提供的各种工具能满足任何项目需要。

虚拟仪器技术将计算机应用于测试仪器之中，利用良好的虚拟仪器软件平台充分发挥计

算机强大的数据处理功能和丰富的图形显示功能。在屏幕上虚拟出与传统仪器相似的显示面板，用户通过键盘和鼠标操纵面板上的虚拟开关、旋钮、按键等，控制仪器的运行、了解仪器的状态、读取测试结果。虚拟仪器以特定的软件支持取代相应的电子线路，充分利用计算机硬件资源，完成传统仪器硬件的部分甚至全部功能。可以说虚拟仪器技术是传统仪器功能和外形的模块化和软件化。

虚拟仪器主要包含两方面的含义：虚拟仪器的面板是虚拟的；虚拟仪器测量功能是通过对图形化软件流程图的编程来实现的。它的主要特点有：

（1）用户可以根据自己的需要定义和制造各种仪器。虚拟仪器通过提供给用户组建自己仪器的可重用源代码库，可以修改仪器功能和面板，设计仪器功能，实现与外设、网络及其他连接。

（2）虚拟仪器尽可能采用通用的硬件，各种仪器的差异主要是软件，突出了“软件就是仪器”的新概念。

（3）虚拟仪器充分利用了计算机强大的数据处理、传输和发布功能，可以创造出功能性很强的仪器，使得组建系统变得更加灵活、简单，从而便于构成复杂的测试系统。

（4）虚拟仪器硬件和软件都制定了开放的工业标准，用户可以将仪器的设计、使用和管理统一到虚拟仪器标准，使得功能更易于扩展，且生产、维护和开发费用降低。

6.2.2 虚拟仪器系统组成

虚拟仪器由通用仪器硬件平台和应用软件两大部分构成。

1. 虚拟仪器的硬件平台

构成虚拟仪器的硬件平台由计算机和 I/O 接口设备两部分组成。计算机一般为一台 PC 机或者工作站，是硬件平台的核心。I/O 接口设备主要完成输入信号的采集、放大、模/数转换。不同的总线有其相应的 I/O 接口设备，如利用 PC 机总线的数据采集卡/板（简称为数采卡/板，DAQ）、GPIB 总线仪器、VXI 总线仪器模块、串口总线仪器等。

虚拟仪器的构成方式主要有 5 种类型，如图 6-1 所示。

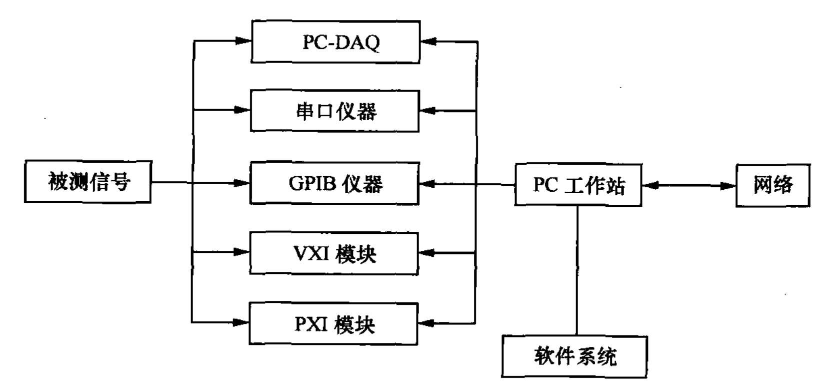


图 6-1 虚拟仪器构成方式图

PC-DAQ 系统是以数据采集板、信号调理电路及计算机为仪器硬件平台组成的插卡式虚拟仪器系统，这种系统采用 PCI 或计算机本身的工业总线，将数据采集卡/板（DAQ）插入计算机的空槽中即可。

串口系统是以 Serial 标准总线仪器与计算机为仪器硬件平台组成的虚拟仪器测试系统。GPIB 系统是以 GPIB 标准总线仪器与计算机为仪器硬件平台组成的虚拟仪器测试系统。VXI 系统是以 VXI 标准总线仪器模块与计算机为仪器硬件平台组成的虚拟仪器测试系统。PXI 系统是以 PXI 标准总线仪器模块与计算机为仪器硬件平台组成的虚拟仪器测试系统。无论上述哪种 VI 系统，都通过应用软件将仪器硬件与计算机相结合。

2．虚拟仪器的软件

开发虚拟仪器必须有合适的软件工具，目前的虚拟仪器软件开发工具只有以下两类。

（1）文本式编程语言：如 Visual C++，Visual Basic，LabWINDOWS/CVI 等；

（2）图形化编程语言：如 LabVIEW、HPVEE 等。

这些软件开发工具为用户设计虚拟仪器应用软件提供了最大限度的方便条件与良好的开发环境。虚拟仪器的软件由应用程序和 I/O 接口仪器驱动程序两部分构成，虚拟仪器的应用程序包含实现虚拟面板功能的软件程序以及定义测试功能的流程图软件程序。

6.2.3　虚拟仪器与传统仪器的比较

虚拟仪器在智能化程序、处理能力、性能价格比、可操作性等方面都具有明显的技术优势，具体表现为：

（1）智能化程度高，处理能力强。虚拟仪器的处理能力和智能化程度主要取决于仪器软件水平。用户完全可以根据实际应用需求，将先进的信号处理算法、人工智能技术和专家系统应用于仪器设计与集成，从而将智能仪器水平提高到一个新的层次。

（2）复用性强，系统费用低。应用虚拟仪器思想，用相同的基本硬件可构造多种不同功能的测试分析仪器，如同一个高速数字采样器，可设计出数字示波器、逻辑分析仪、计数器等仪器。这样形成的测试仪器系统功能更灵活、系统费用更低。通过与计算机网络连接，还可实现虚拟仪器的分布式共享，更好地发挥仪器的使用价值。

（3）可操作性强。虚拟仪器面板可由用户定义，针对不同应用可以设计不同的操作显示接口。使用计算机的多媒体处理能力可以使仪器操作变得更加直观、简便、易于理解，测量结果可以直接进入数据库系统或通过网络发送。测量完后还可打印，显示所需要的报表曲线，这些都使得虚拟仪器的可操作性大大提高。

虚拟仪器与传统仪器的比较见表 6-1。

表 6-1　　虚拟仪器与传统仪器比较表

仪器类型 / 项目比较	传统仪器	虚拟仪器
关键要素	硬件	软件
技术更新周期	长	短
仪器功能定义	厂商	用户
系统	封闭、固定	开放、灵活
与其他设备连接情况	困难	容易
开发维护费用	高	低
价格	高	低

6.3 虚拟仪器的开发环境介绍

数据采集、仪器控制、过程监控和自动测试是实验室研究和工业自动化领域广泛存在的实际任务。在20世纪80年代个人计算机出现之前，几乎所有拥有程控仪器的实验室都采用贵重的仪器控制器测试系统，这些功能单一、价格昂贵的仪器控制器通过一个集成通信口来控制IEEE-488总线仪器（GPIB程控仪器）。后来随着PC的出现，工程师和科学家们找到了一种性能价格比高的通用PC控制台式仪器的方法，虚拟仪器开发工具应运而生。

虚拟仪器的开发工具有多种，如LabWINDOWS/CVI，LabVIEW，HPVEE等，其中LabVIEW使用最广泛。本节首先讲述了LabVIEW的基本概念，然后分析了用LabVIEW创建一个VI的基本步骤，给出了一个用LabVIEW创建VI的示例，最后对LabVIEW的使用给出了一些总结。读者需要细细体会，并将本例实践一下，方可做到对LabVIEW的入门。

6.3.1 LabVIEW简介

LabVIEW（Laboratory Virtual Instrument Engineering Workbench）的概念雏形是来源于美国国家仪器公司（NATIONAL INSTRUMENTS）的特鲁查德和柯德斯凯，在20世纪70年代末期，他们在应用研究实验室（Applied Research Laboratory）完成了一个大型测试系统。该系统主要用于测试美国海军的声呐探测器。通过几年的时间，柯德斯凯把从该测试系统得到的基本思想发展到测试系统软件可以由多层虚拟仪器（Virtual Instruments，VI）构成的新概念。一个VI可以由更低层的多个VI组成，就像真实仪器由印制电路板组成，而印制电路板又由集成电路（IC）组成一样。底层VI代表了最基本的软件功能——计算与输入/输出（I/O）操作。虚拟仪器模型的另一个主要特征是每一个VI都有一个用户接口组件（VI前面板）。同时，在分析比较了几种框图编程方法的优劣后，柯德斯凯决定采用数据流程图作为编程工具。柯德斯凯领导的开发小组于1986年5月推出LabVIEW Beta测试版。又经过几个月的反馈修改，于1986年10月正式发布了LabVIEW1.0版。1988年的LabVIEW2.0采用了面向对象编程技术，1992年8月LabVIEW2.5实现了从Macintosh平台到Windows平台的移植。从LabVIEW3.0版本开始，LabVIEW作为一个完整优异的图形化软件开发环境得到了工业界和学术界的认可，并开始迅速占领市场，赢得了广大用户的青睐。

LabVIEW的基本特点有：

（1）具有良好的用户接口。其用户接口类似于传统仪器的面板，包括按钮、旋钮、图形显示组件、控制组件等。通过鼠标和键盘向程序输入数据，操作结果由软件在计算机屏幕上生成。

（2）编程方式简单、直观，采用图形语言（G语言）、图标和联机代替文本形式编写程序，是对具体编程问题的图形化解决方案。

（3）具有层次结构和模块化的特点，每一个VI可以作为顶层程序，也可以作为其他程序的子程序。

（4）提供具有程序调试功能的程序调试工具，包括在源代码中可以设置断点，可以单步

执行，也可以启动。

6.3.2 LabVIEW 创建 VI 的基本过程

基于 LabVIEW 创建虚拟仪器的过程大致分为 4 步。

（1）创建前面板。前面板是图形化用户界面，用于设置输入数值和观察输出量。它模仿了实际仪器的面板。前面板包含了旋钮、按钮、图形和其他控制与显示对象。通过鼠标和键盘输入数据、控制按钮，也可在计算机显示器上直接观看结果。若想要在数字控制中输入或修改数值，只需要用操作工具单击控制部件和增减按钮，或者用操作工具或标签工具双击数值栏进行输入数值修改。

（2）创建框图程序。在前面板窗口的主菜单“窗口”中选择“显示程序框图”将前面板窗口切换到框图程序窗口，此时会看到与前面板对象对应的端口。根据需要在功能模板中找到所需的节点，并将节点图标放置到框图程序窗口。用数据连线将这些端口和节点的图标连接起来，形成一个完整的框图程序。

（3）创建图标。一个虚拟仪器的图标/连接端口就像一个图形（表示某一虚拟仪器）的参数列表。这样，其他的虚拟仪器才能将数据传输给下一个子仪器。图标和连接允许将此仪器作为最高级的程序，也可以作为其他程序或子程序的子程序。

（4）调试和运行程序。运行和调试程序是任何一门编程语言编程的最重要的一步。如果一个 VI 程序存在语法错误，则在面板工具条上的运行按钮将会变成一个折断的箭头，表示程序不能被执行。这时这个按钮被称作错误列表。单击它，则 LabVIEW 弹出错误清单窗口，单击其中任何一个所列出的错误，选用“寻找”功能，则出错的对象或端口就会变成高亮。调试程序时可以利用单步执行、设置断点、设置探针来显示数据流动方向。

6.3.3 利用 LabVIEW 创建 VI 的示例

本节详细介绍 LabVIEW 中的前面板，程序框图等模块的使用，其中给出了使用中的一些使用技巧。此例来源于 NI 公司所提供的练习，详细的描述了创建一个 VI 的过程，该 VI 可产生信号并在图形中显示信号。开发软件使用的是 NI LabVIEW2009，读者可以到 NI 官方网站下载，或者购买光盘安装。读者按照如下步骤进行操作，即可熟悉 LabVIEW 的基本组件，完成一个 VI 的创建，最终可完成如图 6-2 所示的 VI 的前面板。

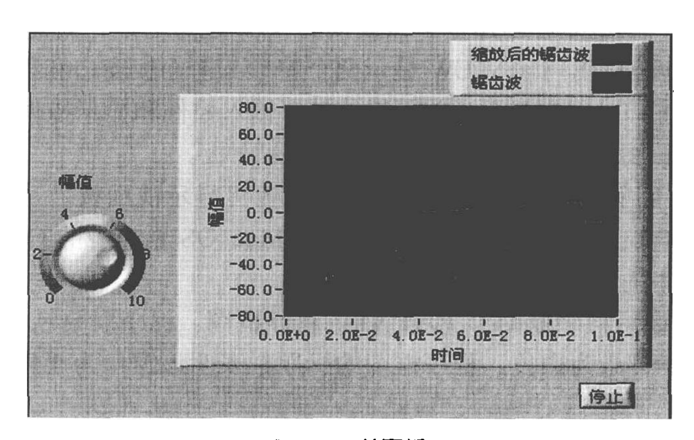


图 6-2 前面板

1. 启动 LabVIEW

启动 LabVIEW 时可显示启动窗口，如图 6-3 所示。通过该窗口中可新建 VI、选择最近打开的 LabVIEW 文件、查找范例以及打开 LabVIEW 帮助。同时还可查看各种信息和资源（例如，用户手册、帮助主题）以及 NI 网站 ni.com 上的各种资源。

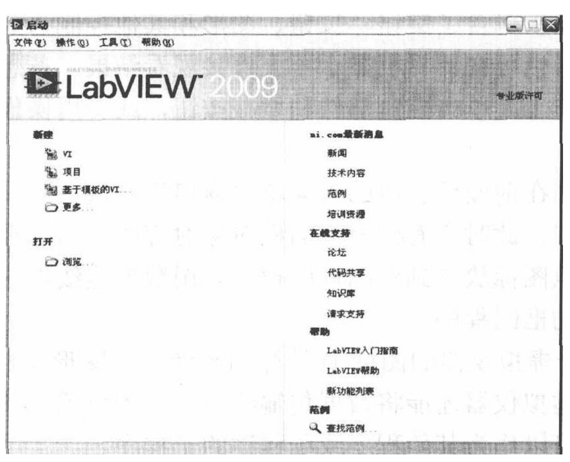


图 6-3　启动窗口

打开现有文件或新建文件后启动窗口将消失。关闭所有已打开的前面板和程序框图后可再次显示启动窗口。在前面板或程序框图窗口中选择“查看/启动窗口”，也可显示启动窗口。

2. 打开基于模板的新 VI

LabVIEW 提供的内置 VI 模板，包含用于创建常规测量应用程序所需的子 VI、函数、结构和前面板对象。按照下列步骤，创建生成信号并在前面板中显示该信号的 VI。

（1）启动 LabVIEW。

（2）在启动窗口中单击新建或基于模板的 VI 链接，可显示新建对话框。

（3）在新建列表中选择“VI/基于模板/使用指南（入门）/生成和显示”。该 VI 模板可生成并显示信号。VI 模板的预览和简要说明位于窗口右侧的说明部分。如图 6-4 所示。为新建对话框和该 VI 模板的预览。

（4）单击“确定”按钮即可创建基于该模板的 VI。也可通过在新建列表中双击 VI 模板的名称创建基于该模板的 VI。LabVIEW 显示两个窗口：前面板窗口和程序框图窗口。

（5）查看前面板窗口。用户界面（前面板，包含输入控件和显示控件）的背景色为灰色。前面板的标题栏表明该窗口为“生成和显示”VI 的前面板。

提示：如前面板不可见，选择“窗口/显示前面板可显示前面板”。按<Ctrl+E>键可切换前面板和程序框图窗口。快捷键中的 Ctrl 键相当于 Mac OS 系统中的 Option 键或 Command 键，或者 Linux 系统中的 Alt 键。

（6）选择“窗口/显示程序框图”，检查 VI 的程序框图。程序框图包含用于控制前面板对象的各种 VI 和结构，背景为白色。程序框图的标题栏表明该窗口为“生成和显示”VI 的程序框图。

（7）单击前面板工具栏上的运行按钮，如图 6-5 所示。也可以按<Ctrl+R>键运行 VI。前

面板上的图形可显示正弦波。

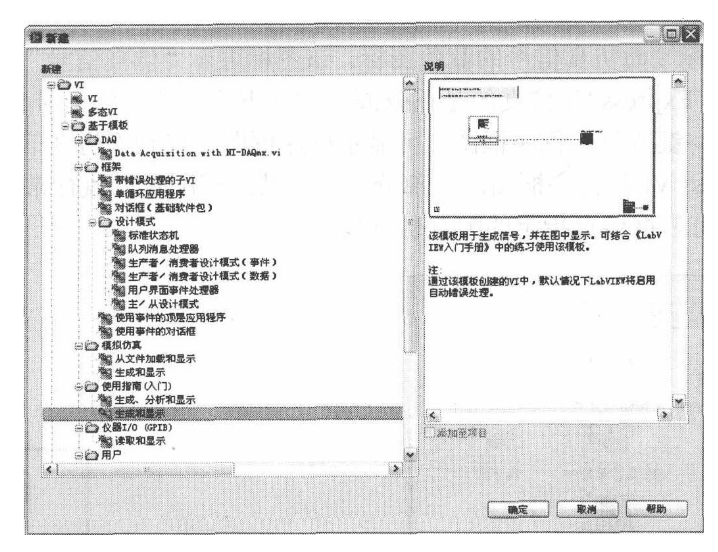


图 6-4　新建对话框和该 VI 模板的预览

（8）如需停止 VI，可单击前面板上的停止按钮，如图 6-6 所示。



图 6-5　运行按钮　　　　图 6-6　停止按钮

3．为前面板添加输入控件

前面板上的输入控件相当于物理仪器的输入装置，为 VI 的程序框图提供数据。许多物理仪器都有旋钮，转动旋钮可改变输入值。按照下列步骤，为前面板添加旋钮输入控件。

提示：在整个过程中，可选择“编辑/撤销”或按<Ctrl+Z>键撤销此前操作。

（1）前面板上未显示控件选板时，可选择“查看/控件选板”。如图 6-7 所示。

提示：右键单击前面板或程序框图的任意空白，也可显示临时的控件或函数选板。控件和函数选板的左上角显示图钉图标，单击该图钉图标可锁定浮动的选板。

（2）默认状态下，初次使用 LabVIEW 时打开控件选板可显示 Express 选板。如图 6-7 所示。如未显示 Express 选板，单击控件选板上的 Express 可显示 Express 选板。

（3）在 Express 选板图标上移动光标，定位在数值输入控件选板。光标在控件选板的图标上移动时，图标下方的提示框可显示光标所在子选板和控件的名称。

提示：某些函数选板对象在选板上显示短名称，可能与提示框中显示的内容不同。短名称是选板对象名称的缩写，适合选板上有限的空间。如通过短名称查找选板对象不方便，可使用控件或函数选板上的搜索按钮，按名称查找选板对象。

（4）单击数值输入控件，可显示数值输入控件选板。

（5）单击数值输入控件选板上的旋钮输入控件，旋钮控件附着在光标上时，添加旋钮至前面板上波形图的左侧。随后的练习中将使用该旋钮控制信号的幅值。

（6）选择“文件/另存为”，将 VI 命名为 Acquiring a Signal.vi，保存在易于访问的位置。

4. 改变信号的类型

程序框图上有标签为仿真信号的蓝色图标。该图标表示“仿真信号”Express VI。默认状态下，“仿真信号”Express VI 仿真的是正弦波。按照下列步骤，将信号改为锯齿波。

（1）按<Ctrl+E>键或单击程序框图，可显示程序框图。找到“仿真信号”Express VI，如图 6-8 所示。Express VI 是程序框图的一部分，可对其进行配置以执行常规测量任务。“仿真信号”Express VI 可依据用户指定的配置仿真信号。

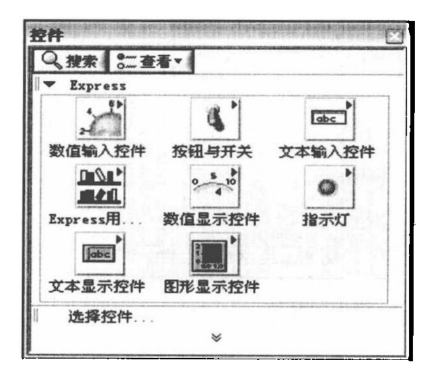


图 6-7 控件选板



图 6-8 仿真信号 Express VI

（2）右键单击“仿真信号”Express VI，在快捷菜单中选择属性，显示配置仿真信号对话框。在 Mac OS 系统中按 Command 键，相当于右键双击该 Express VI，也可显示配置仿真信号对话框。如连线数据至 Express VI 并运行 VI，该 Express VI 可在配置对话框中显示实际数据。如关闭后重新打开 Express VI，配置对话框中将显示示例数据，直至再次运行时才显示实际数据。

（3）在信号类型下拉菜单中选择锯齿波。

结果预览区域中显示的波形为锯齿波。图 6-9 所示为配置仿真信号对话框。

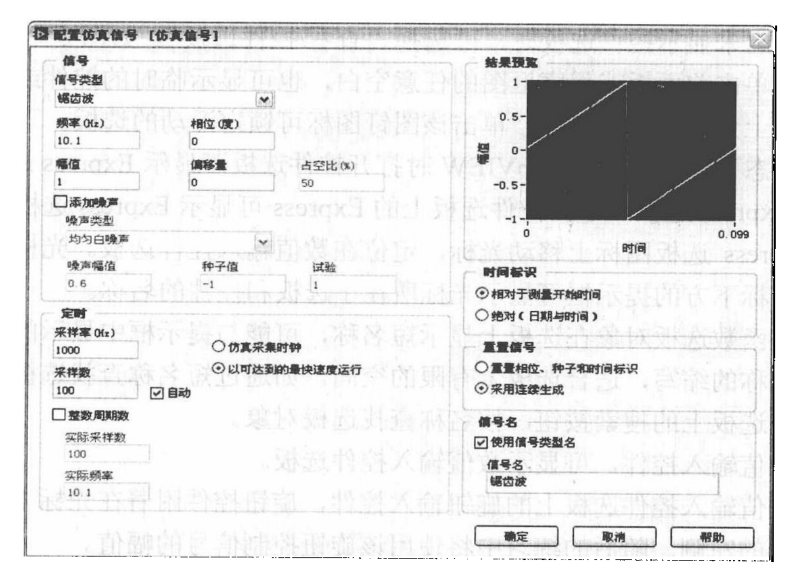


图 6-9 配置仿真信号对话框

（4）单击确定按钮，保存当前配置并关闭配置仿真信号对话框。

（5）移动光标至“仿真信号”Express VI 下方的下拉箭头。拖动 Express VI 的下拉箭头，可显示隐藏的输入和输出端。

（6）显示双箭头时，单击双箭头并将 Express VI 的边框向下拖拽两行，如图 6-10 所示。释放光标，显示幅值输入端，可在程序框图上配置锯齿波的幅值。图 6-10 中的幅值是配置仿真信号对话框中的选项。程序框图上显示输入端（例如，幅值），且在配置对话框中有对应选项时，可选择任意位置配置该输入。



图 6-10 输入端设置

5．连线程序框图上的对象

如需通过旋钮更改信号的幅值，必须连线程序框图上的两个对象。按照下列步骤，连线旋钮和“仿真信号”Express VI 的幅值输入端。

（1）在程序框图上，移动光标至旋钮的接线端上方，如图 6-11 所示。此时光标显示为箭头（定位工具）。如图 6-12 所示。定位工具用于对象的选择、定位或调整大小。

提示：可在程序框图上调整循环或结构的大小。可在前面板上调整对象的大小。

（2）通过定位工具选定旋钮接线端，置于“仿真信号”Express VI 的左侧且位于灰色循环结构的内部，如图 6-13 所示。循环内的接线端分别表示前面板上的输入控件和显示控件。接线端是前面板和程序框图之间交换信息的输入/ 输出端口。

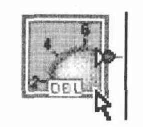


图 6-11 旋钮

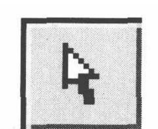


图 6-12 箭头

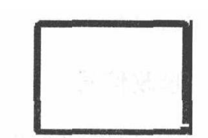


图 6-13 灰色循环结构

（3）单击程序框图中的空白，可取消选定旋钮接线端。如需在对象上使用其他工具，必须先取消选定对象，才可切换工具。

（4）移动光标至旋钮接线端的箭头上方，如图 6-14 所示。光标显示为线圈（连线工具），如图 6-15 所示。连线工具用于连接程序框图上的对象。

（5）显示连线工具时，单击旋钮接线端的箭头，再单击“仿真信号”Express VI 幅值输入端的箭头，可连线两个对象，如图 6-16 所示。此时，两个对象间显示连线。数据通过该连线从旋钮接线端传递至 Express VI。

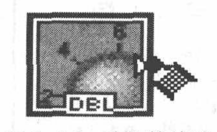


图 6-14 旋钮接线端

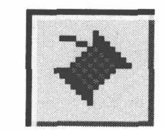


图 6-15 线圈（连线工具）

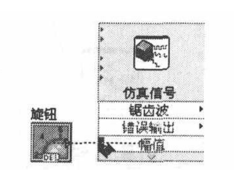


图 6-16 连线两个对象

（6）选择文件/保存，保存 VI。

6．运行 VI

运行 VI 可执行程序，按照下列步骤，运行采集信号 VI。

（1）按<Ctrl+E>键或单击前面板可显示前面板。

（2）单击运行按钮或按<Ctrl+R>键可运行 VI。

（3）移动光标至旋钮控件上方，光标显示为手形（操作工具），如图 6-17 所示。操作工具用于改变控件的值。

（4）通过操作工具转动旋钮可调节锯齿波的幅值。转动旋钮时，锯齿波的幅值随之改变，更改幅值时，操作工具在提示框中显示旋钮的数值。图形的 Y 轴可根据幅值的改变自动调整标尺。运行按钮显示为黑色箭头时，表示 VI 正在运行，如图 6-18 所示。VI 运行时可更改绝大多数输入控件的值，但是无法编辑 VI。



图 6-17 手形



图 6-18 黑色箭头

（5）单击停止按钮可停止 VI 运行，如图 6-19 所示。停止按钮可在 VI 完成当前循环后停止 VI 的运行。单击“中止执行”按钮，可在 VI 完成当前循环前立即停止 VI 的运行，如图 6-20 所示。中止使用外部资源（例如，外部硬件）的 VI 可能导致外部资源无法恰当复位或释放并停留在未知状态，设计 VI 时添加停止按钮可避免此类问题。



图 6-19 停止按钮

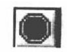


图 6-20 中止执行按钮

7. 修改信号

按照下列步骤，将信号缩放 10 倍并在前面板上的图形中显示结果。

（1）在程序框图上，通过定位工具双击连接“仿真信号”Express VI 和波形图接线端的连线，如图 6-21 所示。

（2）按<Delete>键可删除该连线。

（3）如未显示如图 6-21 所示的 Express VI 和波形图接线端连线图和图 6-22 所示的函数选板，通过选择“查看/函数选板”打开函数选板，默认显示 Express 选板。如已选择其他选板，在函数选板上单击“Express”按钮，即可返回 Express 选板。

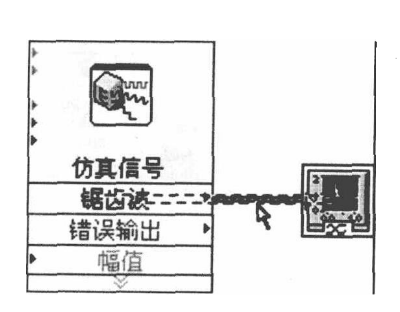


图 6-21 Express VI 和波形图接线端连线图

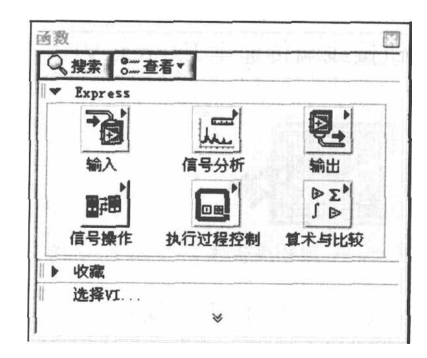


图 6-22 函数选板

（4）在算术与比较选板上选择“公式”Express VI，如图 6-23 所示。放置在循环内，位

于“仿真信号”Express VI 和波形图接线端之间。适当右移波形图接线端，使 Express VI 与接线端之间有更多空间。“公式”Express VI 放置于程序框图上时，可自动显示配置公式对话框。通常在程序框图上放置 Express VI 时，可自动显示该 VI 的配置对话框。

提示：如程序框图上放置的对象间距过小，自动连线功能可连线相邻的对象。应删除错误的自动连线。选择“工具/选项”，在类别列表中选择程序框图。取消勾选启用自动连线复选框，可禁用自动连线。

（5）单击配置公式对话框右下角的帮助按钮，显示 LabVIEW 帮助中该 Express VI 的帮助主题，如图 6-24 所示。公式的帮助主题介绍了该 Express VI 配置对话框选项以及 Express VI 的输入和输出。每个 Express VI 都有相应的帮助主题，单击 Express VI 配置对话框中的帮助按钮，或者右键单击“Express VI”按钮，在快捷菜单中选择“帮助”，可查看相关帮助主题。

（6）通过公式的帮助主题中对话框选项的说明，应为公式输入变量。

（7）最小化 LabVIEW 帮助窗口，返回配置公式对话框。

（8）依据帮助主题中对话框选项的说明，更改标签列中的 X1 为锯齿波，可指定公式 Express VI 的输入值，如图 6-25 所示。单击配置公式对话框的公式文本框，文本更改为输入的标签。



图 6-23　公式 Express V

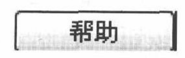


图 6-24　帮助主题

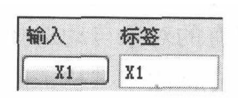


图 6-25　输入的标签

（9）在公式文本框的锯齿波后输入*10，指定缩放因子的值。配置缩放因子时，可使用配置对话框中的输入按钮，也可使用键盘上的*、1 和 0 直接输入。如使用配置对话框中的输入按钮，LabVIEW 将在公式文本框中的锯齿波后放置输入的公式。如使用键盘直接输入，单击锯齿波后的公式文本框，可输入公式。图 6-26 所示为配置公式对话框。

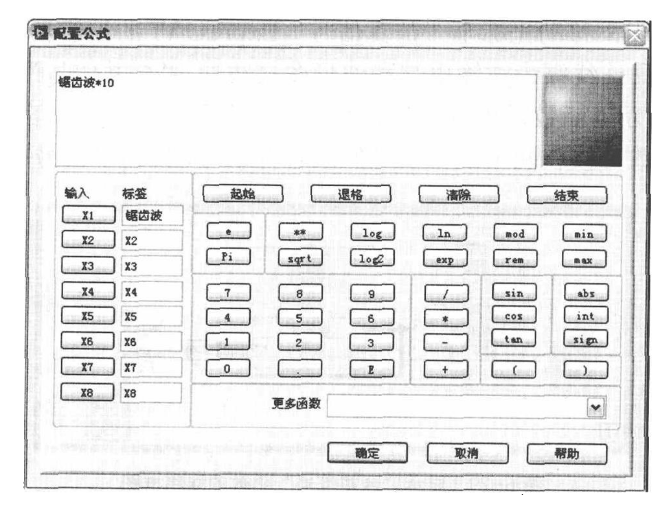


图 6-26　配置公式对话框

提示：如在公式文本框中输入的公式为非法公式，则右上角的错误指示灯将由绿变灰并显示该非法公式。

（10）单击“确定”按钮，保存当前配置并关闭配置公式对话框。

（11）移动光标移至“仿真信号”Express VI 的锯齿波输出端的箭头上方。

（12）显示连线工具时，单击锯齿波输出端的箭头，再单击“公式”ExpressVI 的锯齿波输入端的箭头，连线两个对象，如图 6-27 所示。

（13）通过连线工具连接“公式”Express VI 的结果输出端和波形图接线端。查看 Express VI 与接线端之间的连线。Express VI 和接线端上的箭头表示连线上数据流的方向。图 6-28 所示为程序框图。

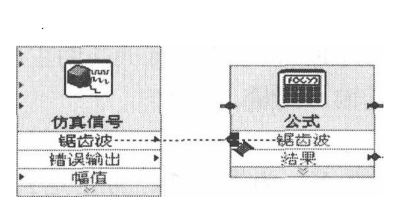


图 6-27　连接两个对象

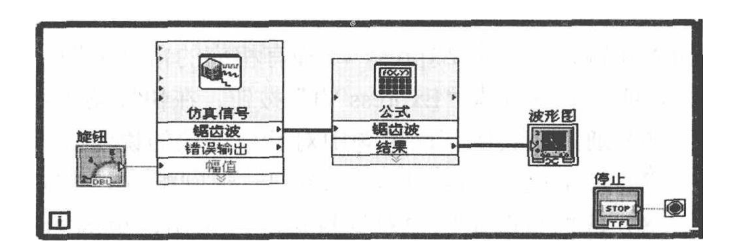


图 6-28　采集信号 VI 的程序框图

提示：右键单击任意连线，在快捷菜单中选择“整理连线”，LabVIEW 可依据程序框图中现有的对象自动选择最佳连线路径。选择路径时，LabVIEW 可自动减少连线转折，也可单击程序框图工具栏上的“整理程序框图”按钮，通过 LabVIEW 自动整理程序框图上已有的连线和对象，获得更清晰的布局。

（14）按<Ctrl+S>键或选择“文件/保存”，可保存 VI。

8．在图形上显示两个信号

如需在同一个图形中比较“仿真信号”Express VI 产生的信号与“公式”Express VI 调整的信号，可使用“合并信号”函数。按照下列步骤，在同一个图形中显示两个信号。

（1）在程序框图上，移动光标至“仿真信号”Express VI 的锯齿波输出端的箭头上方。

（2）通过连线工具连线锯齿波输出端和波形图接线端。“合并信号”函数位于两条连线的连接处，如图 6-29 所示。函数是内置的执行元素，相当于文本编程语言中的运算符、函数或语句。“合并信号”函数可接收两个独立信号然后合并信号，使两个信号在同一个图形中显示。

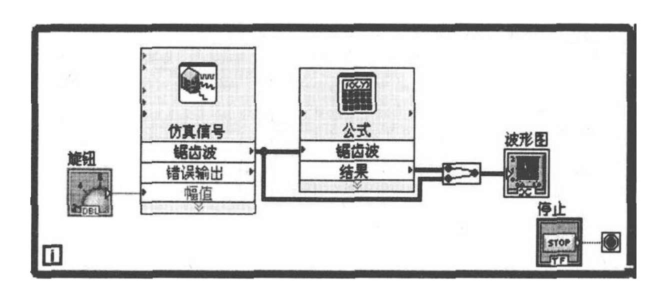


图 6-29　显示“合并信号”函数的程序框图

（3）按<Ctrl+S>键或选择“文件/保存”，可保存 VI。

（4）返回至前面板，运行 VI，转动旋钮控件。依据“公式”Express VI 指定的配置，图形可显示原有锯齿波和幅值增大 10 倍后的锯齿波。转动旋钮控件时，y 轴的最大值可自

动缩放。

（5）单击“停止”按钮，中止 VI 运行。

9. 自定义旋钮输入控件

旋钮输入控件用于更改锯齿波的幅值，使用幅值标签可更准确描述旋钮的作用。按照下列步骤，自定义旋钮的外观。

（1）在前面板上，右键单击旋钮，在快捷菜单中选择属性，显示旋钮属性对话框，如图6-30 为旋钮属性对话框。



图 6-30 旋钮属性对话框

（2）在外观选项卡上的标签区域，删除旋钮标签，输入幅值。

（3）单击标尺选项卡，勾选标尺样式区域的显示颜色梯度控件复选框。前面板上的旋钮可显示相应更新。

（4）单击“确定”按钮，保存当前配置并关闭旋钮属性对话框。

（5）保存 VI。

（6）重新打开旋钮属性对话框，尝试旋钮的其他属性。例如，在标尺选项卡上，单击颜色盒可更改标记文本颜色。

（7）单击“取消”按钮，可避免应用所做的改动。如需保存所作的修改，可单击“确定”按钮。

10. 自定义波形图

波形图显示控件显示了两个信号。对曲线进行自定义，可区分缩放信号和仿真信号的曲线。按照下列步骤，自定义波形图显示控件的外观。

（1）在前面板上，移动光标移至波形图图例的顶端。虽然图形中有两条曲线，但图例中仅显示一条曲线。

（2）显示双箭头时，单击并拖动图例边框，使图例显示第二条曲线，如图 6-31 所示。释放鼠标后，可显示第二条曲线的名称。

（3）右键单击波形图，在快捷菜单中选择“属性”，显示图形属性对话框。

图 6-31 展开图例

（4）在曲线选项卡上，在下拉菜单中选择锯齿波。在颜色区域，单击线条颜色盒，显示颜色选择器。选择新的线条颜色。

（5）在下拉菜单中选择锯齿波（公式结果）。

（6）在名称文本框中，删除当前标签，更改曲线名称为缩放后的锯齿波。

（7）单击“确定”按钮，保存当前配置并关闭图形属性对话框。前面板上曲线的颜色和图例已改。

（8）重新打开图形属性对话框，尝试图形的其他属性。例如，在标尺选项卡上，可尝试禁用自动调整标尺，更改 Y 轴的最大值和最小值。

（9）单击“取消”按钮，可避免应用所做的改动。如需保存所作的修改，可单击“确定”按钮。

（10）保存并关闭 VI。

6.3.4 利用 LabVIEW 创建 VI 的总结

1. 新建对话框和 VI 模板

新建对话框包含许多 VI 模板（也包括本手册使用的模板）。VI 模板用于帮助用户创建用于常规测量和其他任务的 VI。VI 模板包括初步创建常规测量应用程序所需的 Express VI、函数和前面板对象。可通过下列任意方法打开新建对话框：

（1）运行 LabVIEW 后，在启动窗口中单击“新建”、“基于模板的 VI...”或“更多...”链接。

（2）在启动窗口、前面板或程序框图的菜单栏中选择“文件/新建”。

2. 前面板

前面板是 VI 的用户界面。输入控件和显示控件是 VI 的交互式输入和输出端口，用于创建前面板。输入控件和显示控件位于控件选板，输入控件是指旋钮、按钮、转盘等输入装置。输入控件模拟仪器的输入装置，为 VI 的程序框图提供数据；显示控件是指图表、指示灯等显示装置。显示控件模拟仪器的输出装置，用于显示程序框图获取或生成的数据。

3. 程序框图

程序框图包含图形化源代码（G 代码或程序框图代码），可确定 VI 的运行方式。程序框

图代码使用图形化表示的函数控制前面板对象。前面板对象在程序框图上显示为图标接线端，通过连线使控件的接线端与 Express VI、VI 和函数连接。数据通过连线由输入控件传递至 VI 和函数，再传递至其他 VI 和函数，最后传递至显示控件。数据在程序框图节点间的传输可确定 VI 和函数的执行顺序，该方式称为数据流编程。

4．前面板和程序框图工具

光标移至前面板或程序框图中的对象时，可显示定位工具。光标显示为箭头，用于对象的选择、定位和调整大小。移动光标至程序框图对象的接线端时，可显示连线工具。此时，光标显示为线圈，用于连接程序框图上的对象，使数据在对象间流动。

5．运行和停止 VI

运行 VI 可执行该 VI 程序。单击“运行”按钮或按<Ctrl+R>键可运行 VI。运行按钮显示为黑色箭头时，表明 VI 正在运行；单击“中止执行”按钮，可立即停止 VI 运行。如 VI 使用外部资源，中止 VI 可能导致外部资源处于未知状态，设计 VI 时添加停止按钮可避免此类问题，停止按钮可在 VI 完成当前循环后停止 VI 的运行。

6．Express VI

函数选板上的 Express VI 用于常规测量任务。在程序框图上放置 Express VI 时，可自动显示 Express VI 的配置对话框。对话框中的各个选项用于指定 Express VI 的行为。也可双击 Express VI 或右键单击“Express VI”，在快捷菜单中选择“属性”，显示配置对话框。如连线数据至 Express VI 并运行 VI，该 Express VI 可在配置对话框中显示实际数据。如关闭后重新打开 Express VI，配置对话框中将显示实例数据，直至再次运行时才显示实际数据。Express VI 在程序框图上可显示为扩展节点，通过调整 Express VI 的大小可显示或隐藏输入或输出。Express VI 显示的输入和输出由具体配置确定。

7．LabVIEW 文档资源

LabVIEW 帮助包括 LabVIEW 编程理论、使用 LabVIEW 的分步指导以及 LabVIEW 中 VI、函数、选板、菜单、工具、属性、方法、事件、对话框等对象的参考信息，LabVIEW 帮助还包括 NI 提供的各种 LabVIEW 文档资源。配置 Express VI 时，单击配置对话框的“帮助”按钮可查看该 Express VI 的帮助信息，也可右键单击程序框图或已锁定函数选板上的 VI 或函数。在快捷菜单中选择帮助，或者选择“帮助/搜索 LabVIEW 帮助”，打开“LabVIEW 帮助”。安装 LabVIEW 附加软件（例如，工具包、模块、驱动程序）后，LabVIEW 帮助可显示附加软件的文档，或者在独立的帮助系统中显示（可通过“帮助/附加软件帮助”打开，“附加软件帮助”是独立的帮助系统的名称）。

8．属性对话框

属性对话框或快捷菜单可用于配置前面板上输入控件和显示控件的外观或行为。右键单击前面板上的控件，在快捷菜单中选择属性，可打开对象的属性对话框。VI 运行时，无法打开控件的属性对话框。

9．快捷键

快捷键中的 Ctrl 键相当于 Mac OS 系统中的 Option 键或 Command 键，或者 Linux 系统中的 Alt 键。

<Ctrl+R>：运行 VI。

<Ctrl+Z>：撤销此前操作。

<Ctrl+E>：在前面板和程序框图窗口之间切换。

<Ctrl+S>：保存 VI。

6.4 虚拟仪器的综合应用举例

本节主要介绍了虚拟仪器技术的两个实际开发应用实例，第一个例子介绍了利用 LabVIEW 开发的智能心音检测仪，该例全面考虑了心音检测的特点，充分利用了 LabVIEW 的优势，开发了一个具有实用价值且有特色的虚拟仪器。第二个例子简要介绍了利用 LabVIEW 开发一个二阶虚拟实验平台，给出了所用到的模块和应用方案，但未给出详细的开发步骤，有兴趣的读者可以查阅相关资料，按照所给思路自行开发出二阶虚拟实验平台。

6.4.1 智能心音检测仪

1．开发目的和意义

传统的心脏听诊技术是以人耳听音来进行的，这种方式受限于人耳听力的灵敏度和主观经验与判断能力，其作用很有限。随着 ECG 和超声多谱勒仪等先进的辅助诊断仪器的开发与利用，心音信号的利用受到冷落。但超声多谱勒仪，因其价格昂贵，并不易于普及。ECG 信号虽然对于血液循环和血液组织相关疾病的诊断比较有效，但是却不能有效地反映与实质性心脏病有关的病理信息。从上述心音的产生机理和心音与心脏瓣膜疾病的关系可知，心音信号中恰恰包含了心脏瓣膜疾病的丰富信息，在检测该类型疾病的领域，心音信号有着无可比拟的优越性。同时心音信号检测方便、无创、花费极小，可作为心脏病检测、预防的常规手段。因此，研制一种能简易、方便地检测心音信号的心音检测分析仪，对于满足医院和病人的需要，有着极大的社会价值和经济价值。

基于此目的，利用虚拟仪器技术，采用多种分析方法对心音信号进行采集、去噪、分析，从多个角度全面了解心音的特性，并为相应心脏疾病的诊断提供有力的依据，为临床应用提供有效的分析手段。

本系统利用 LabVIEW 语言作为开发工具，结合 MATLAB 小波工具箱以及 MATLAB 强大的数学计算能力，以 Windows 为操作系统，开发一个智能心音检测仪，实现对心音的采集、分析、处理一体化，为临床应用提供了一个基本的平台，并将在进一步的临床研究中，不断完善整个系统的结构，强化系统的功能，使其成为一个集医疗、研究、教学等多种功能为一体的，具有较高性价比的实用仪器。

2．虚拟心音信号采集和分析系统的总体设计

智能心音检测仪分为心音信号采集子系统、小波去噪子系统和心音信号分析子系统 3 个子系统，系统结构如图 6-32 所示。

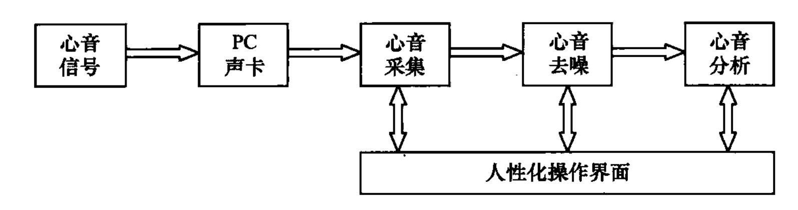


图 6-32 系统结构

（1）系统的硬件构成。

① 性能良好的普通个人台式机或笔记本电脑，要求带有能正常工作的声卡。

近年来声卡成长快速，目前各类品牌的声卡非常多，但是就声卡的一些基本原理来看，其实都是一样的。一般来说，人耳的听力范围在 20Hz 到 20kHz 之间，因此，只要采样频率达到 20kHz×2=40kHz 时，就可以满足人们的要求。现时大多数声卡的采样频率都已达到 44.1kHz 或 48kHz，即达到所谓的 CD 音质水平了。

② 心音传感器。

采用新型高分子聚合材料制作的心音传感器采集来自心脏的心音信号，再经过放大处理，经声卡将心音信号送入计算机。该检测装置的放大倍数为 10～1000 倍自动调整，灵敏度 4m V/P a，过载能力为 100 倍。由于采样频率对采集信号的质量影响明显，故在系统中设计了采样频率可调的采集模式。因为心音低频端处于人耳听阈以外，所以只有用心音检测仪才能真实地显示心音的波形特性。

（2）系统开发软件平台。

① LabVIEW 8.6 程序开发软件。

LabVIEW 8.6 是美国国家仪器有限公司推出的 LabVIEW 图形化开发平台的新版本，提高了设计、控制和测试领域工程师的效率，同时包括对 LabVIEW 实时模块、LabVIEW FPGA 模块、LabVIEW PDA 模块以及 LabVIEW 数据记录和监控模块的升级。LabVIEW 8.6 提供大量完成信号产生、信号处理、信号分析及滤波器设计的子程序，结合普通 PC 声卡可以方便的采集声音信号。

② Matlab 7.0 语言编译平台。

Matlab 是一种面向科学与工程计算的高级语言，目前已发展成为国际上先进的科技应用软件之一。它拥有强大的科学计算功能、完整的数字信号处理和图形图像工具箱支持，使系统的即时性和准确性得到有力的保证。

（3）LabVIEW 与 Matlab 程序接口。

LabVIEW 提供了 Matlab 的接口，这实际上就是通过 Active X 控件与 Matlab 进行通信。通过这种方式，用户可在 LabVIEW 中使用 Matlab 强大的数值运算功能，但这种方法不能脱离 Matlab 环境，而只是将它在后台执行。

在 LabVIEW 程序框图设计窗口下，移动光标到程序框图设计器区，单击鼠标右键，打

开“函数”选板，从中选择“数学”选板下的“脚本与公式”函数选板，打开该选板中的“脚本节点”，如图 6-33 所示，可以看到 MATLAB 脚本节点位于该选板中。

从脚本节点选板对象中，选择“MATLAB 脚本节点”放置到程序框图设计区，如图 6-34 所示，该脚本节点是一个空的方框图，没有“输入/输出”节点。根据用户的需求，移动光标到脚本节点框图边缘上，单击鼠标右键，弹出如图 6-34 所示的快捷菜单，执行“添加输入”或“添加输出”菜单命令，可以为该节点创建“输入/输出”端口，并且编辑创建的端口名称。



图 6-33 "脚本节点"选板对象

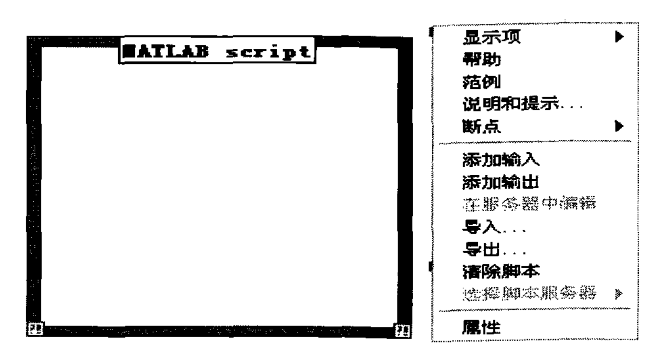


图 6-34 MATLAB 脚本节点与脚本节点快捷菜单

（4）子 VI 节点的建立及调用。

LabVIEW 可将新创建的 VI 用于另一个 VI。一个 VI 被其他 VI 在程序框图中调用，则称该 VI 为子 VI，子 VI 可重复调用。要创建一个子 VI，需先为子 VI 创建连线板和图标。子 VI 的节点类似于文本编程语言中的子程序调用。一个程序中的子程序调用指令并不是子程序本身，同理，节点也不是子 VI。一个程序框图含有相同子 VI 节点的数目与该子 VI 被调用的次数相等。

子 VI 的控件和函数从调用该 VI 的程序框图中接收数据，并将数据返回至该程序框图。如需创建一个被调用的子 VI，单击函数选板上的选择 VI 图标或文本，找到目标 VI 并双击，即可将该 VI 放置在程序框图上。

用操作或定位工具双击程序框图上的子 VI，即可编辑该子 VI。保存子 VI 时，子 VI 的改动将影响到所有调用该子 VI 的程序，而不只是当前程序。

LabVIEW 调用子 VI 时，该子 VI 仅运行而不显示前面板。如希望某个子 VI 在被调用时显示前面板，右键单击该 VI 并从快捷菜单中选择设置子 VI 节点。如希望每个子 VI 实例在被调用时都显示前面板，选择“文件/VI 属性”，从类别下拉菜单中选择窗口外观，单击“自定义”按钮，选择“调用时显示前面板”。

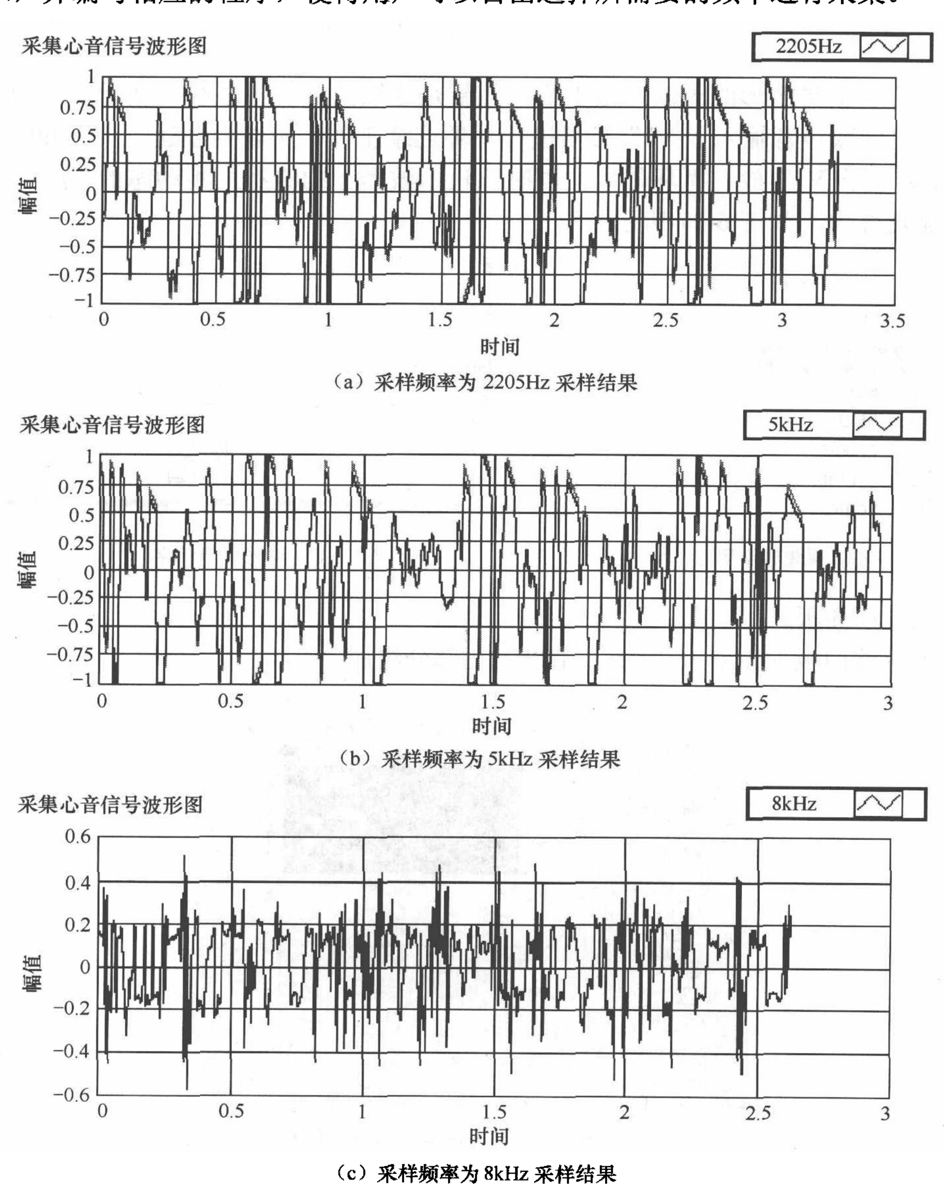
3. 心音信号采集子系统的开发与设计

为了让计算机能够准确、高效地获取被测心音信号，编制了相应的数据采集程序，指定模拟信号的输入路线并按规定的各相关参数通过数据采集卡（PC 声卡）进入计算机。实时处理数据的目的是确保实验安全、加速实验进程和缩短实验周期，而程序具有事后处理数据的能力，可以使用户待实验结束后能对全部数据做完整、详尽的分析。为了使该子系统具有实时显示测量数据和事后处理数据的能力，即能在实验过程中显示心音信号波形，便于现场实时观察分析，及时判断实验对象的状态和性能以及保存心音信号，方便后续研究，本子系统实现了心音信号实时显示和存储功能。

（1）数据采集及显示部分。

在采集心音信号数据时，分析了心音信号的特点，其主要成分的频率在500Hz以下，心脏杂音在1500Hz以下。为了便于辨别效果，选择在噪声相对较大的环境下，分别用频率为2205Hz、5kHz、8kHz、11.025kHz进行采样，并且导出的采集效果图进行比较，如图6-35（a）、（b）、（c）、（d）所示。

经比较发现，在噪声环境下，采集频率越低，心音信号受噪声的干扰越大，而随着采集频率的提高，采集到的第一心音与第二心音越发明显，这将有利于后续的包络提取等研究，另外考虑到心音采集存在着采集频率与系统资源之间的矛盾，所以选择11.025kHz作为心音信号采集子系统的默认采集频率。为了满足不同用户对采集频率需求，添加了输入控件如图6-36所示，并编写相应的程序，使得用户可以自由选择所需要的频率进行采集。

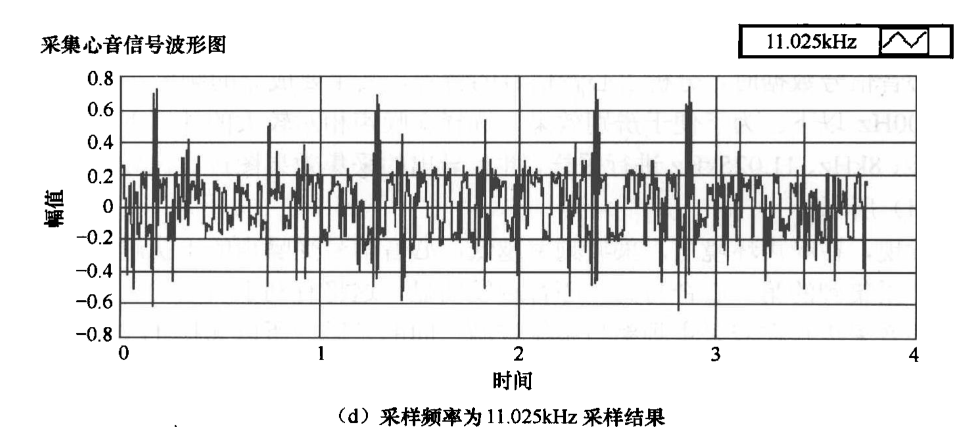


（a）采样频率为2205Hz采样结果

（b）采样频率为5kHz采样结果

（c）采样频率为8kHz采样结果

图6-35　不同采样频率采样结果对比图



（d）采样频率为11.025kHz采样结果

图 6-35　不同采样频率采样结果对比图（续）

在显示心音信号波形的时候，由于心音信号是连续的，而本系统是基于实时的采集心音信号，所以“读取声音输入子 VI”必须进行循环连续工作，所以要创建一个 While 循环，把“读取声音输入子 VI”放在循环体内，同时要让示波器上的波形同步动态的显示，示波器也须放在该循环体内，如图 6-37 所示。

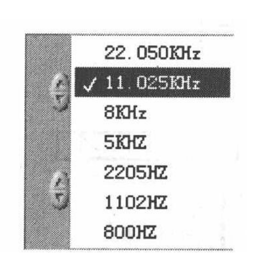


图 6-36　用户选择采集频率界面

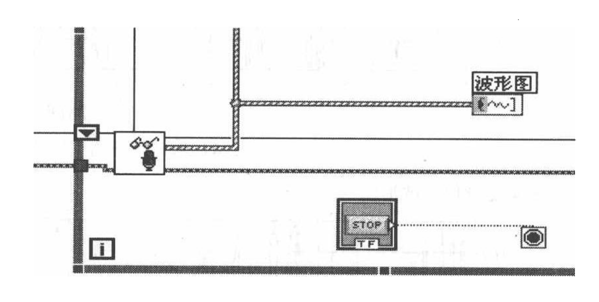


图 6-37　波形显示程序

（2）系统前面板。

本子系统前面板如图 6-38 所示，由以下几部分组成。

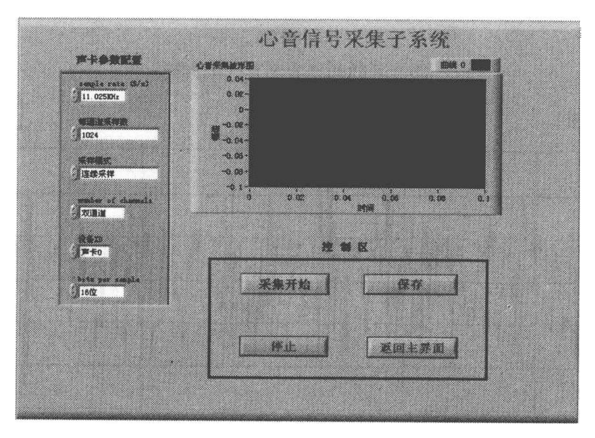


图 6-38　心音信号采集子系统前面板

① 声音格式参数配置部分：声音格式参数的配置包括系统对模拟心音信号的采样率

(S/s)，心音信号通道数以及每采样比特数的设置等，用户通过配置这些参数来满足实际采样需要。

② 心音采集波形显示器部分：用户可以通过原始信号波形显示器直观的看到采集到的心音信号的波形状态。

③ 控制部分：控制部分包括开始采集、保存、停止以及返回主界面 4 个控件，其中保存心音按钮能让用户实时保存采集到的心音。

（3）LabVIEW 程序部分。

程序流程图如图 6-39 所示。

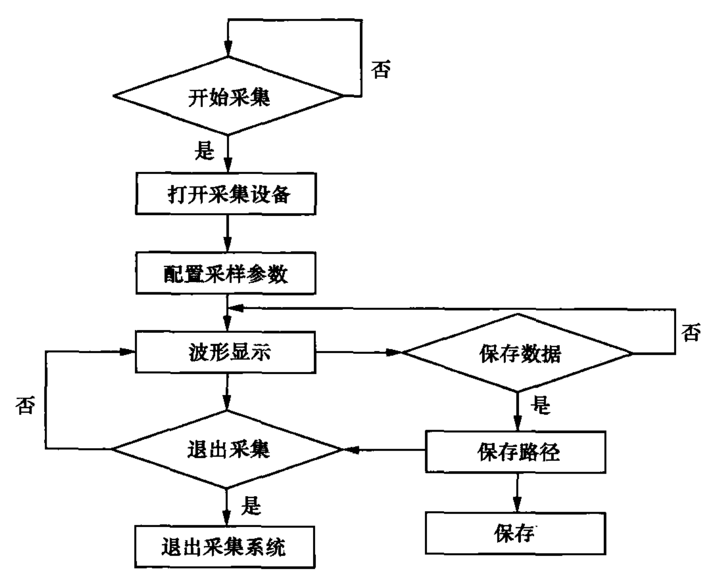


图 6-39 图程序流程图

主要驱动控制部分包括声音格式(Sound Format)参数设置簇、配置声音输入(Sound Input Configure)子 VI、启动声音输入采集（Sound Input Start）子 VI、读取声音输入（Sound Input Read)子 VI、停止声音输入采集(Sound Input Stop)子 VI 以及声音输入清零(Sound Input Clear)子 VI 几部分组成。

声音格式（Sound Format）参数设置簇（图 6-40）：由一系列的输入控件组成，负责从用户处得到用户输入的参数，并传递给配置声音输入函数。

配置声音输入（Sound Input Configure）函数（图 6-41）：配置声音输入设备，采集数据并将数据发送到缓存。



图 6-40 声音格式（Sound Format）参数设置簇

图 6-41 配置声音输入（Sound Input Configure）函数

启动声音输入采集（Sound Input Start）函数（图 6-42）：开始从用户所选设备上采集数据，只有停止声音输入采集函数被调用时才能使用该 VI。

读取声音输入（Sound Input Read）函数（图 6-43）：从声音输入设备读取数据。必须使用“配置声音输入”VI 来配置设备。必须手动选择所需多态实例。



图 6-42 启动声音输入采集（Sound Input Start）函数

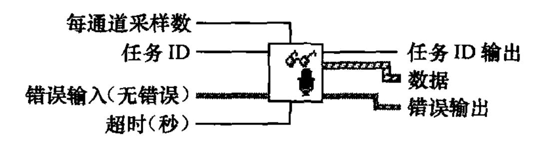


图 6-43 读取声音输入（Sound Input Read）函数

停止声音输入采集（Sound Input Stop）函数：停止从设备采集数据。使用声音输入清零 VI，清除缓存中的数据。使用启动声音输入采集 VI，在调用“停止声音输入”VI 后重新开始采集，如图 6-44 所示。

声音输入清零（Sound Input Clear）函数（图 6-45）：使设备停止播放音频，清空缓存，将任务返回至默认的未配置状态，并释放与任务相关的资源。

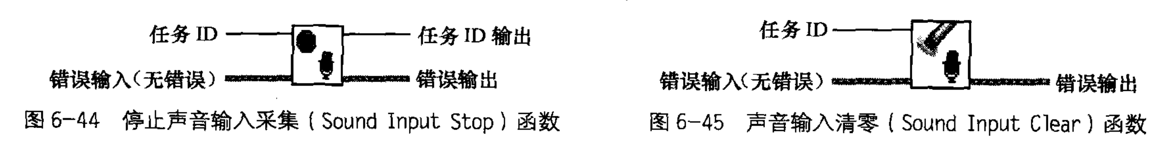


图 6-44 停止声音输入采集（Sound Input Stop）函数　　图 6-45 声音输入清零（Sound Input Clear）函数

4．心音小波去噪子系统的开发与设计

心音是微弱低频生理信号，可以通过心音采集器来进行采集，因此混入噪声是不可避免的，其中噪声也是复杂多样的，一般情况下主要有：随机噪声、仪器噪声、工频干扰几种，为了尽量降低被测心音信号中的噪声成分，系统通过 LabVIEW 平台的数学公式节点调用 MATLAB 的小波工具箱对信号进行去噪，最终实现对信号的预处理。

该子系统在实现对心音信号进行小波去噪预处理功能的同时，还有数据保存功能，能将去噪后的数据保存至文件，便于进一步的研究。

（1）心音小波去噪子系统前面板。

心音小波去噪子系统前面板如图 6-46 所示。心音小波去噪子系统包括：原始信号波形显示器，去噪后波形显示器，小波参数设置，执行小波去噪按钮，保存去噪后数据按钮以及返回主界面按钮。

原始信号波形及去噪后信号波形显示器部分：显示原始信号以及去噪后信号的波形，并且能够让用户直观的看到去噪前后的心音对比。

小波参数设置：经过上面的讨论得出，采用 coif5 小波进行 6 层小波分解为最佳方法，所以把以上参数设置为默认值。但是用户在实际应用研究中会对小波参数有不同的要求，所以用户可以在本界面通过更改小波参数满足相应的应用要求。

小波去噪按钮：对已选择的文件执行小波去噪分析。

保存分析后数据按钮：将去噪后的波形数据保存在用户指定的路径下，便于后续的研究。

（2）程序流程图。

程序流程图如图 6-47 所示。

心音小波去噪子系统的 LabVIEW 程序主要由三个事件结构组成，分别为：执行小波去噪事件，保存去噪后数据事件，退出心音去噪事件以及结合了 MATLAB Script 节点的小波去

噪子程序构成。

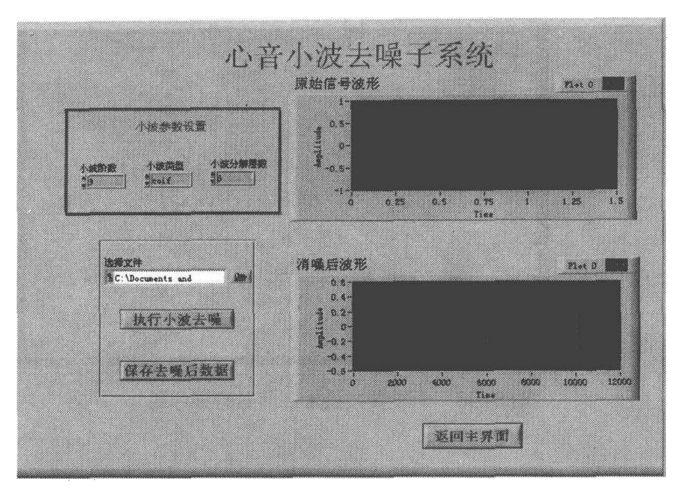


图 6-46　心音小波去噪子系统前面板

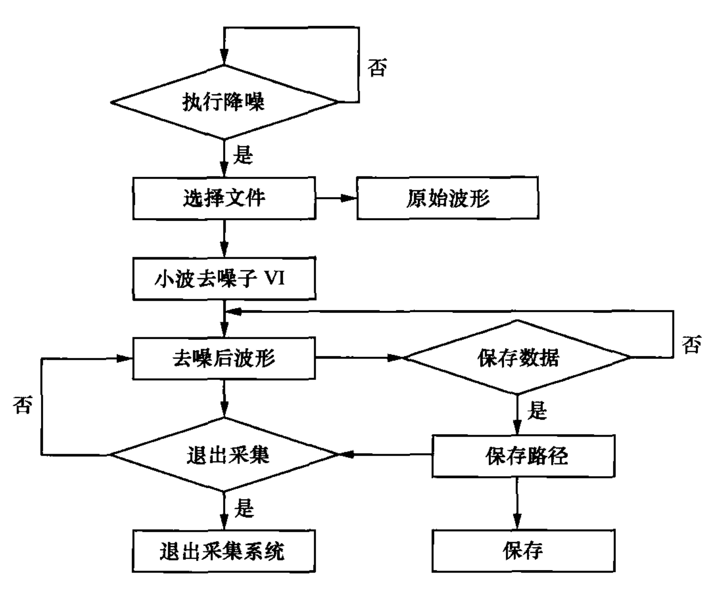


图 6-47　心音小波去噪子系统程序流程图

① 3 个事件结构中最主要的事件是执行小波去噪事件，它包括了打开和读取文件的系统 VI，设置小波参数的条件结构（图 6-48）以及结合 MATLAB script 节点的小波去噪子 VI。

② 小波去噪子 VI 如图 6-48、图 6-49 所示，主要负责心音小波去噪，首先接受从调用它的主程序传来的小波参数，然后传给 MATLAB Script 节点，由 MATLAB 负责小波分析部分，最后再把分析后的值传至 LabVIEW 做进一步的处理。

5．心音分析子系统的开发与设计

心音信号是非平稳信号，为全面了解心音信号的特性，就需要研究心率特性以及信号的时频特性。此处简略介绍频域分析，同时还将计算提取心音信号的 FFT 幅度谱与功率谱，将其显示在前面板上提供给用户，有兴趣的读者可进行时域分析。

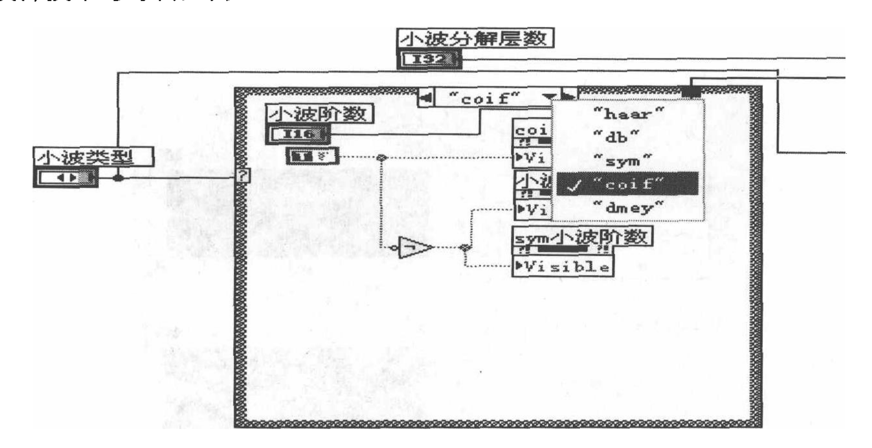


图 6-48　小波参数设置条件结构程序图

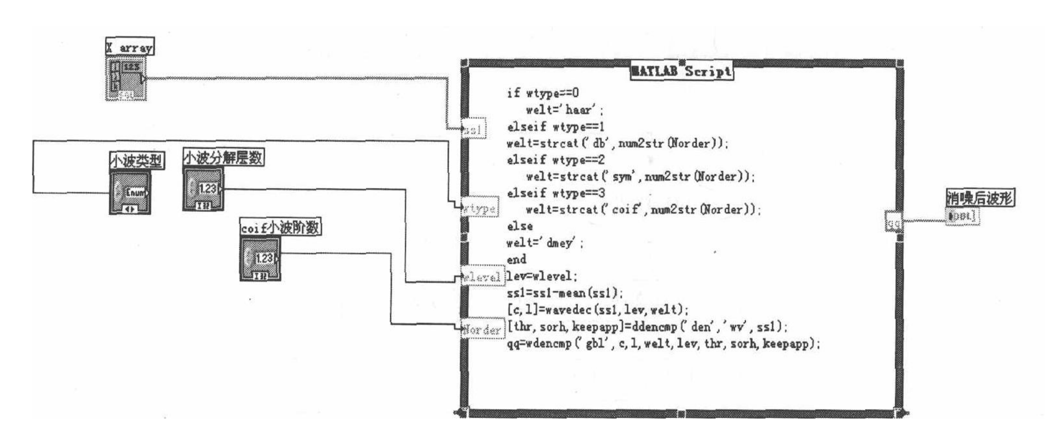


图 6-49　小波去噪子 VI 程序图

（1）频域分析。

频域分析部分主要包括计算信号的 FFT 频谱以及信号的功率谱密度函数。FFT 频谱函数（图 6-50），该函数计算“时间信号”的平均 FFT 频谱，以“幅度”和“相位”返回 FFT 值。FFT 频谱（幅度—相位）VI 按照下列步骤计算“幅度”和“相位”。

① 计算“时间信号”的 FFT。

② 将“时间信号”的当前 FFT 频谱与 VI 自上次平均过程重置后的最后一次计算得到的 FFT 频谱进行平均。

③ 返回平均频谱的幅度和相位。

该 VI 的单通道版本可在单次模式（一次调用）和连续模式（多次调用）下进行单通道测量。单通道 VI 只可在单次模式下进行多通道测量。如需在连续模式下进行多通道测量，应使用该 VI 的多通道版本。本子系统采用的是自动识别输入信号来确定使用的多态实例。

FFT 功率谱 VI（图 6-51）计算“时间信号”的平均自功率谱。它按照下列步骤计算“功率谱”：

① 计算“时间信号”的 FFT。

② 计算“时间信号”的 FFT。

③ 将当前的功率谱与上次平均过程重新开始后的最后一次调用 VI 的功率谱进行平均。

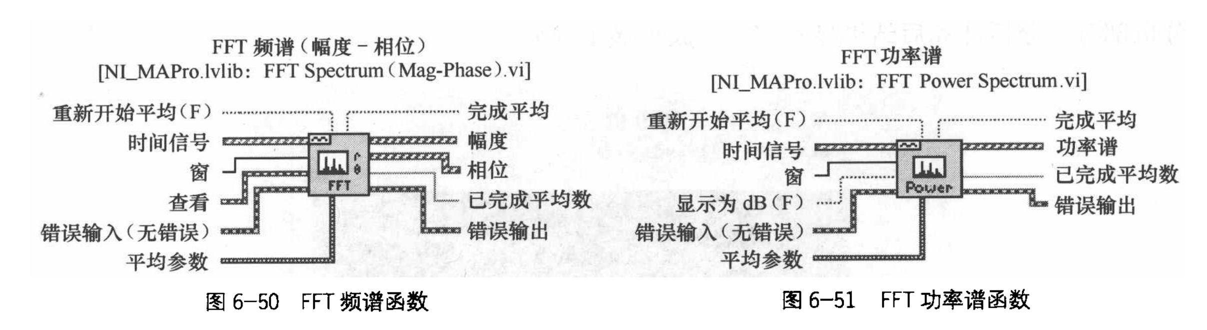


图 6-50　FFT 频谱函数

图 6-51　FFT 功率谱函数

④ 在“功率谱”中返回平均功率谱。

该 VI 的单通道版本可在单次模式（一次调用）和连续模式（多次调用）下进行单通道测量。单通道 VI 只可在单次模式下进行多通道测量。如需在连续模式下进行多通道测量，应使用该 VI 的多通道版本。图 6-52 所示为对一个采样频率为 11025Hz 的心音信号进行 FFT 及 FFT 功率谱计算图。

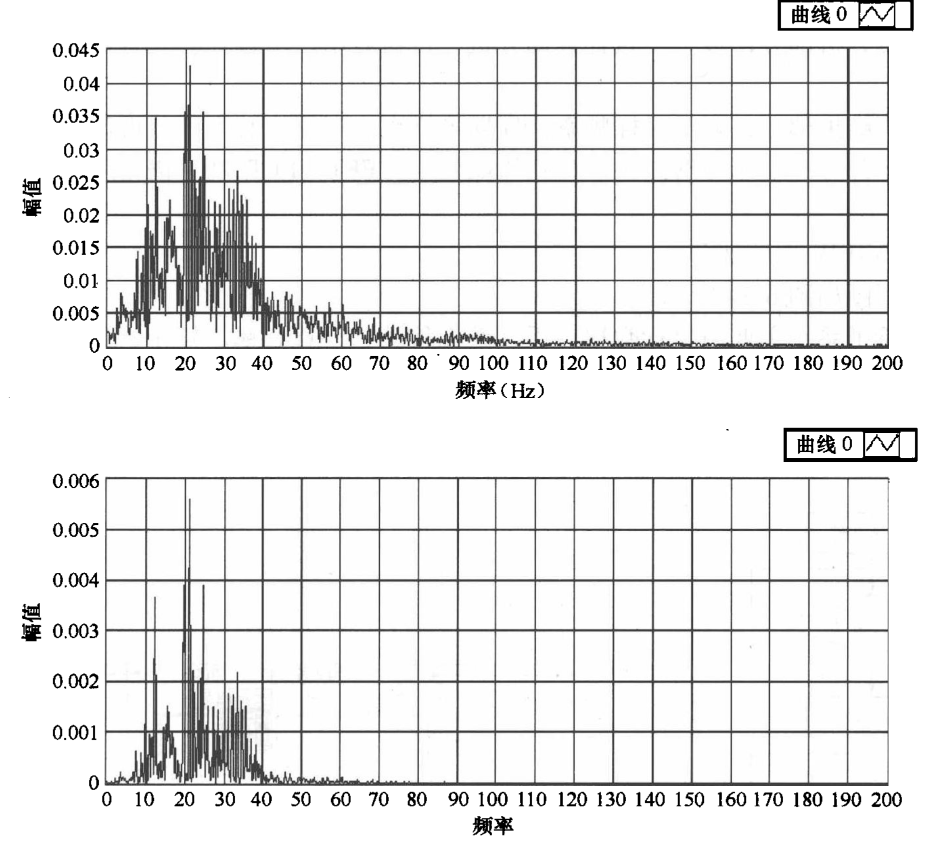


图 6-52　心音信号的 FFT 幅度谱及功率谱图

从 FFT 功率谱图中可以看到，心音信号主要集中在 0～100Hz，功率主要集中在 0～60Hz，在 20Hz 附近最强。

（2）心音信号分析子系统前面板。

心音信号分析子系统前面板如图 6-53 所示。心音分析子系统前面板主要包括：执行数据

分析部分、分析计算后结果显示部分、波形显示部分。

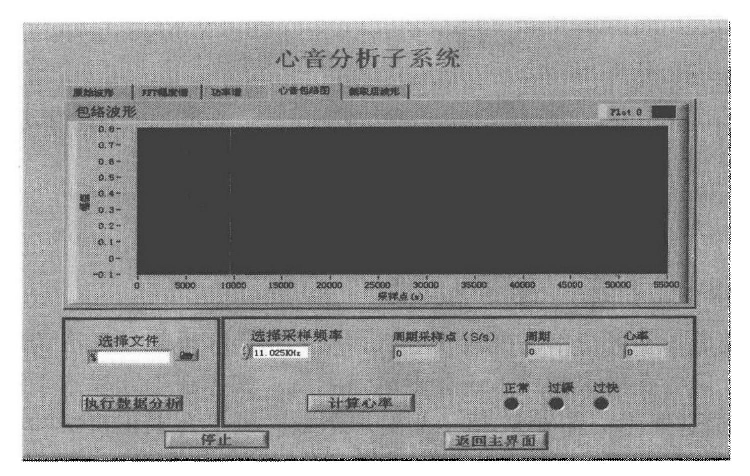


图 6-53　心音信号分析子系统

执行数据分析部分：按下“执行数据分析”按钮对用户所选择的心音信号文件执行数据分析；

分析计算结果部分：显示采样频率，周期采样点，周期，心率等分析计算后的结果；

波形显示部分：显示分解结果，包括原始波形，FFT，FFT 功率谱密度，心音包络图，截取后波形。

（3）LabVIEW 程序部分。

程序流程图如图 6-54 所示。

心音分析子系统主要由执行数据分析、游标移动、计算心率、停止、返回主界面等几个事件结构构成。

执行数据分析事件结构事件包括读取声音信号部分、信号的包络提取以及频谱分析等，当该事件被触发，声音信号将被各个分析模块处理，并输出结果。主要程序如图 6-55 所示。

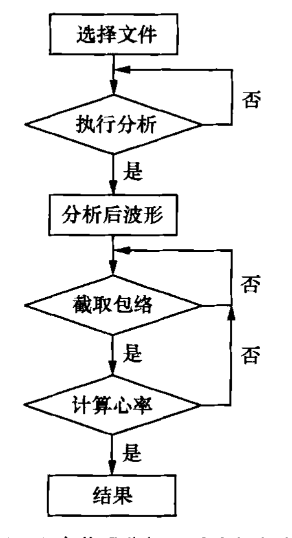


图 6-54　心音信号分析子系统程序流程图

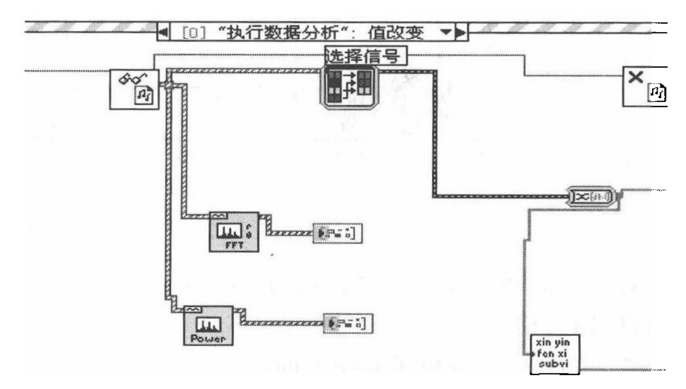


图 6-55　“执行数据分析”事件结构部分程序图

游标移动事件结构当负责截图的游标滑动的时候被触发，活动游标属性节点负责获得游标 0 和游标 1 的位置，然后送到最大值最小值比较函数得出最小值作为截取的起始节点，再将两个游标之差作为截取长度，输入数字子集函数，即可得出用户想要截取的波形，并自动放大。主要部分程序如图 6-56 所示。

计算心率事件结构根据信号的周期采样点数以及信号的采集频率计算心率，然后通过逻辑比较得出心音正常或者非正常，并通过指示灯在前面板提示用户。主要程序部分如图 6-57 所示。

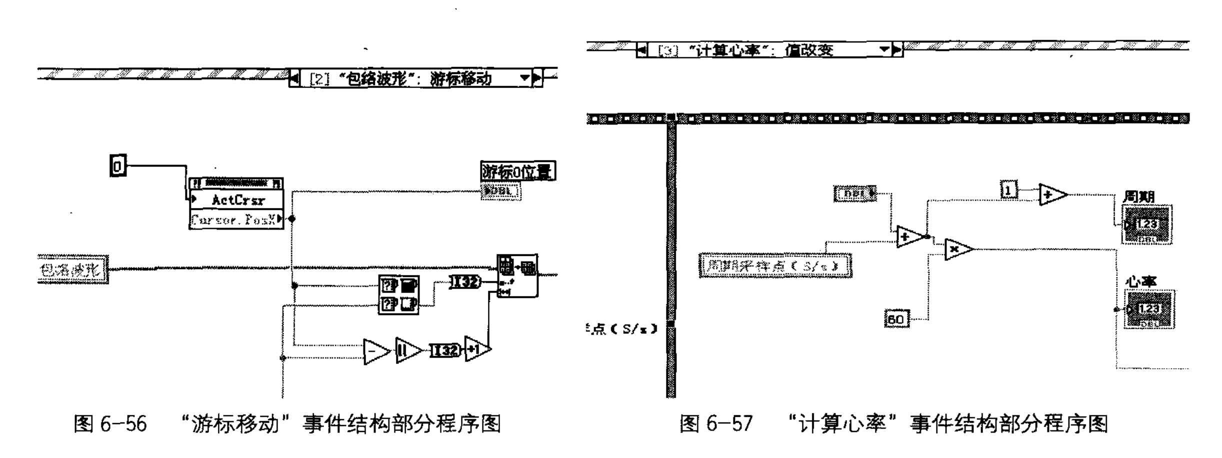


图 6-56　“游标移动”事件结构部分程序图　　图 6-57　“计算心率”事件结构部分程序图

6.4.2　基于 LabVIEW 的二阶系统虚拟实验平台

建立了均具有二阶特性的质量—弹簧—阻尼系统和液压位置随动系统的虚拟实验平台，可调整系统参数，观察典型输入下的系统响应的动态过程，可通过正弦信号的扫频过程研究系统的频率响应特性。可作为控制原理课程的多媒体教学素材应用于课堂上概念的讲解与演示，也可作为学生课外的研究性学习平台，研究二阶系统运动规律，提高学习的积极性。使用 LabVIEW、LabVIEW 控制设计工具包和 LabVIEW 仿真模块开发本系统。

二阶系统在经典控制理论中占有非常重要的地位，许多基本概念，包括时域分析、频域分析、性能指标等，都可通过对二阶系统的分析得到清楚的解释，高阶系统分析在一定条件下也可近似为二阶系统问题。通过二阶系统实验进一步加深对控制理论的基本概念的理解是一种常用的教学手段，本设计基于 LabVIEW 开发了二阶系统虚拟实验平台，包括以下两个部分的内容：弹簧—质量—阻尼虚拟实验，液压位置伺服系统虚拟实验。

弹簧—质量—阻尼虚拟实验对象是用 Picture 控件实时绘出的，由于该系统是具有一个自由度的平动系统，因此根据质量块位移量参数的变化将响应效果动态地显示出来，实现动画显示。利用控制设计工具包计算不同激励下的系统响应，可观察虚拟对象的响应过程，响应曲线和 Bode 图的绘制过程。

在 LabVIEW 建立的弹簧—质量—阻尼虚拟实验和液压位置伺服系统虚拟实验对象如图 6-58 和图 6-59 所示。

LabVIEW 简单易学的图形化编程界面，丰富美观的控件，广泛的硬件支持和大量的算法工具包，不仅在测控行业被广泛接受，而且在理工科实验教学设备方面也得到越来越多的应用。该平台可以作为多媒体教学素材应用于课堂上概念的讲解与演示，也可作为学生研究性

学习的平台在课外进行参数的调整、动态过程观察、基本概念的理解等。

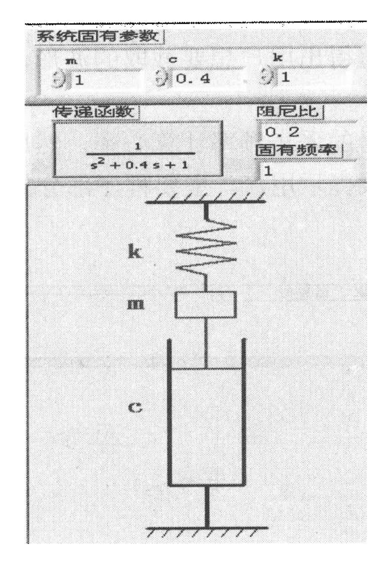


图 6-58　弹簧—质量—阻尼系统虚拟实验对象

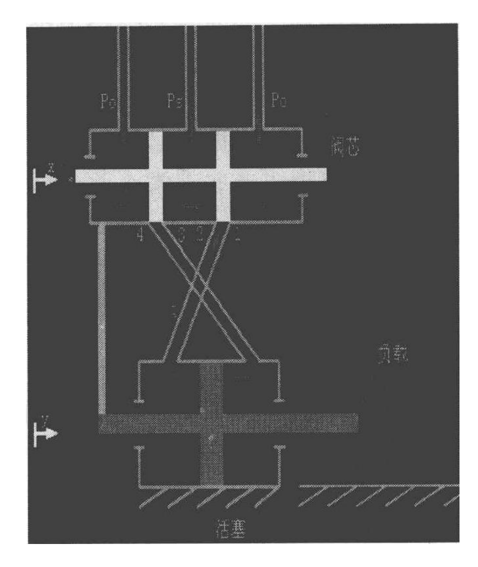


图 6-59　液压位置伺服系统虚拟实验对象

习　题　六

1．什么是虚拟仪器？

2．简述虚拟仪器的优点。

3．简述 LabVIEW 创建虚拟仪器的基本步骤。

4．在 LabVIEW 中有哪三种用来创建和运行程序的摸板？它们都有哪些用途？

5．VI 子程序的连接端口的作用是什么？如何来定义 VI 子程序的连接口？

6．虚拟仪器通用测试平台由哪几个部分组成？它们主要又包括哪些部分？

7．创建一个 VI 程序，该程序可以产生一个六行四列的二维数组（数组元素为 0 到 10 的随机整数），并把二维数组的前三行、前三列单独提出来，做为一个新的数组。

8．用 0-100 的随机数代替摄氏温度，将每 500ms 采集的温度的变化波形表示出来，并设定上下限，温度高于上限或者低于下限分别点亮对应的指示灯，并将其上下限也一并在波形中表示出来。

第7章 现代电子电路设计范例

本章要点

- 现代电子电路的组成及其特点
- 信号获取：常用传感器和输入电路
- 信号处理：转换、合成和分解电路
- 信号输出：显示电路以及电源电路
- 信号执行：基本控制电路
- 现代电子电路设计范例

现代电子技术的设计比传统的电子技术设计有了一些显著的变化，如器件更新更快，电路更依赖于软件，更注重节能与环保。随着无线传感网的发展，给现代电子电路带来很多便利之处的同时也带来一些新的挑战，信号传输的速度越来越快，精度越来越高，这就对电路的设计创新提出了更高的要求。

一个电子系统一般可分为如下几个部分：传感器输入、放大、滤波、采样/保持、A/D 转换、微处理器、D/A 转换、V/I 转换、放大驱动、执行机构等，也可以简单的概括为信号获取、预处理、信号处理、信号执行这几部分。

现代电子设计的特点：

（1）模拟电路仍然是重要的组成部分。要测量的物理量大多以模拟量的形式存在，所以系统中的模拟电路一般必不可少，特别是输入电路、信号调理和输出电路部分。

（2）数字化是信号处理和电子系统设计的大趋势。对于规模较大的数字电路，固定的中、小规模器件几乎被可编程逻辑器件（CPLD/FPGA 等）所代替，以微处理器（CPU、MCU、DSP）为核心的嵌入式系统，已成为系统中控制和信号处理的核心。

（3）电子设计总趋势是：硬件设计软件化，软件设计模块化，软硬件协同工作。

7.1 信号获取电路

信号获取主要是通过传感器，将外界的信息变化为电信号。通过传感器获取的信号往往是比较微弱的信号，需要通过放大电路放大后才能作进一步处理。本节主要介绍一些常用的

传感器以及测量放大器和信号采集电路。

7.1.1 常用的物理传感器及其特性

传感器（Sensor）是感受规定的被测量的各种量并按一定规律将其转换为有用信号（一般为电信号）的器件或装置。对于传感器来说，按照输入的状态划分，输入可以分成静态量和动态量。传感器的静态特性的主要指标有线性度、迟滞、重复性、灵敏度和准确度等。传感器的动态特性指的是对于输入量随着时间变化的响应特性。动态特性通常采用传递函数等自动控制的模型来描述。通常，传感器接收到的信号都有微弱的低频信号，有时候外界干扰信号的幅度超过被测量的信号，因此消除串入的噪声就成为了一项关键的传感器技术。

物理传感器利用某些物理效应，把被测量的物理量转化成为便于处理的能量形式的信号，其输出的信号和输入的信号有确定的关系。主要的物理传感器有光电式传感器、压电传感器、压阻式传感器、电磁式传感器、热电式传感器以及光导纤维传感器等。

1．光电式传感器

光电式传感器是基于光电效应的传感器，在受到可见光照射后即产生光电效应，将光信号转换成电信号输出。光电式传感器除能测量光强之外，还能利用光线的透射、遮挡、反射、干涉等测量多种物理量，如尺寸、位移、速度、温度等，是一种应用极广泛的重要敏感器件。其特点是非接触、响应快、性能可靠。光电测量时不与被测对象直接接触，光束的质量又近似为零，在测量中不存在摩擦且对被测对象几乎不施加压力。因此在许多应用场合，光电式传感器比其他传感器有明显的优越性。其缺点是在某些应用方面，光学电子器件价格较贵，并且对测量的环境条件要求较高。

2．压电传感器

压电传感器是基于压电效应的传感器，是一种自发电式和机电转换式传感器。它的敏感元件由压电材料制成。压电材料受力后表面产生电荷，此电荷经电荷放大器和测量电路放大和变换阻抗后就成为正比于所受外力的电量输出。它的优点是频带宽、灵敏度高、信噪比高、结构简单、工作可靠以及重量轻等，特别适合有很宽频带的周期作用力和高速变化的冲击力的检测。缺点是某些压电材料需要防潮措施，而且输出的直流响应差，需要采用高输入阻抗电路或电荷放大器来克服这一缺陷。配套仪表和低噪声、小电容、高绝缘电阻电缆的出现，使压电传感器的使用更为方便。压电传感器广泛应用于工程力学、生物医学、电声学等技术领域。

3．压阻式传感器

压阻式传感器是利用单晶硅材料的压阻效应和集成电路技术制成的传感器。单晶硅材料在受到力的作用后，电阻率发生变化，通过测量电路就可得到正比于力变化的电信号输出。压阻式传感器用于压力、拉力、压力差和可以转变为力的变化的其他物理量（如液位、加速度、重量、应变、流量等）的测量和控制。压阻式传感器的优点是：频率响应高（如有的产品固有频率在 1.5MHz 以上），适于动态测量；体积小（如有的产品外径可达 0.25mm），适于微型化；精度高，可达 0.01%；灵敏高，比金属应变计高出很多倍，有些应用场合可不加放大器；无活动部件，可靠性高，能工作于振动、冲击、腐蚀、强干扰等恶劣环境。其缺点是

温度影响较大（有时需进行温度补偿）、工艺较复杂和造价高等。

4. 电磁传感器

电磁传感器是通过磁电作用将被测量（如振动、转速、扭矩）转换成电势信号的传感器。它是利用导体和磁场发生相对运动而在导体两端输出感应电动势的，因此它是一种机—电能量变换型传感器，具有不需要供电电源、电路简单、性能稳定、输出阻抗小的优点，能够应用在很大的温度范围中，有很长的工作寿命、抗灰尘、水和油污的能力强，能耐受各种环境条件及外部噪声。

5. 热电式传感器

热电式传感器可以分为两种，热电阻式传感器和热电偶式传感器。

热电阻式传感器是利用导体和半导体材料的电阻率随温度变化的特性制成的传感器，它主要用于对温度和与温度有关的参量进行检测，测温范围主要在中低温区域（−200℃～650℃）。

热电阻式传感器按测量元件可分为金属热电阻和半导体热敏电阻两大类。热敏电阻的温度系数比金属大，电阻率大，可以制成极小的电阻元件，体积小，热惯性小，适用于测量点温、表面温度及快速变化的温度。热敏电阻结构简单、机械性能好，可根据不同要求，制成各种形状。热敏电阻的最大缺点是线性度较差，只在某一较窄温度范围内有较好的线性度，由于是半导体材料，其复现性和互换性较差。

热电偶传感器是一种将温度变化转换为电势变化的传感器。在工业生产中，热电偶是应用最广泛的测温元件之一。其主要优点是测温范围广，可以在 1000℃～2800℃的范围内使用，精度高，性能稳定，结构简单，动态性能好，把温度转换为电势信号便于处理和远距离传输。

6. 光导纤维传感器

光导纤维传感器（简称光纤传感器）是 20 世纪 70 年代迅速发展起来的一种传感器。光纤传感器较传统的传感器相比有许多特点：灵敏度高、结构简单、体积小、耗电量少、耐腐蚀、绝缘性好、光路可弯曲、便于实现远调。从传感器的机理上来说，光纤传感器可分为振幅型（强度型）和相位型（干涉型）两种。

（1）振幅型光导纤维传感器。

振幅型光导纤维传感器通过待测的物理扰动与光纤连接的光纤敏感元件相互作用，直接调制光强。这类传感器的优点是结构简单、具有与光纤技术的相容性，信号检测也比较容易，但是灵敏度较低。

（2）相位型光导纤维传感器。

相位型光导纤维传感器的机理是在一段单模光纤中传输的相干光，因待测物理场的作用，产生了相位调制。理论上，相位型传感器的灵敏度要比现有的传感器高出几个数量级，并可通过改变光纤上的涂层来改变其传感的物理量。缺点是相位型光导纤维传感器结构复杂，检测也需要复杂的手段，需要研制对某种物理量敏感的特种光纤。

当然还有很多的物理传感器，如超声波传感器、红外传感器、激光传感器等，在这不作

详细介绍。

7.1.2 心音传感器及放大电路

心音传感器是心音采集系统的重要组成部分，其作用是将胸壁的机械振动转换成电信号，以方便对信号进行放大以及后续的分析与处理等。心音传感器的类型主要包括空气传导式、接触传导式和加速度式等种类。

空气传导式心音传感器是利用心脏搏动时通过胸壁传递出的音波信号再经空气传到与换能器相连的传感器敏感振动膜上，因而空气振动则膜片产生振动，从而产生与心音强度成比例的输出信号。空气传导型心音传感器可采用电磁感应式、压电式和电容式等原理设计制作。

加速度式心音传感器是采用将低量程高灵敏度的加速度传感器置于胸壁上进行心音信号检测。加速度传感器重量轻、尺寸小、抗干扰能力强、频率响应范围可达 10～800Hz，甚至更高，是目前应用较广的心音传感器类型之一。

接触传导式心音传感器的原理是将胸壁传导出来的心音波动信号直接通过敏感元件传递到换能元件，并转换为电信号，实现心音信号检测。该类型传感器由于结构上没有采用空气作为传递媒介，抵抗外界声波干扰的能力比气导式传感器要好。

根据要获取信号的特点，自己设计制作实用的传感器，是现代电子技术设计发展的一种发展趋势。下面介绍一种获得中国发明专利的“双听诊头的两路心声检测装置”。该装置如图 7-1 所示。基于人体心脏听诊原理和相关的信号处理技术，为了有效提取心音的细微差异，提高心音身份识别率，该装置将一个膜型听诊头和一个钟型听诊头用支架固定成一体，钟型听诊头的位置比膜型听诊头的位置靠后 5mm 左右，两个听诊头谐振腔的顶部通过一段塑料管分别与两个驻极体压电转换器紧密连接。在支架的顶部靠近膜型听诊头的位置安放了一个压力开关，当选好听诊区，用力按下压力开关按钮，使弹簧压紧时，膜型听诊头是紧紧压在皮肤上，钟型听诊头则刚好轻轻扣在皮肤上，同时开关也正好接通。钟型听诊头有利于收集第三、第四心音和来自于二尖瓣、三尖瓣的舒张期杂声；膜型听诊头有利于收集第一、第二心音、收缩期喀喇声和高调杂声。该装置能有效的同时提取两路人体心音信号。

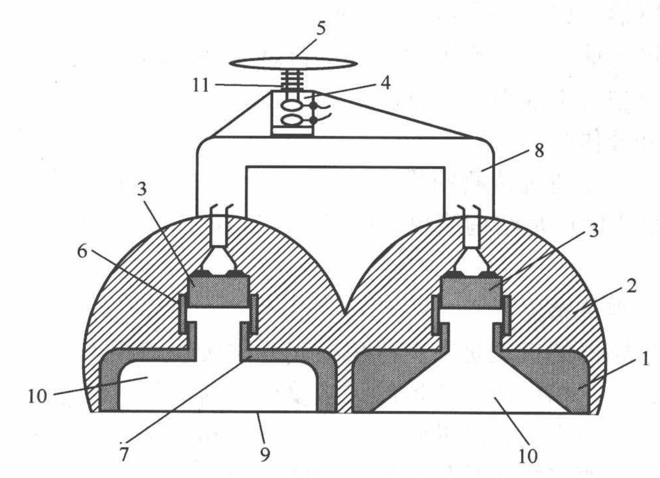


图 7-1　双听诊头的两路心声检测装置的示意图（1 钟型听诊头；2 隔离噪声的泡沫材料；3 驻极体压电转换器；4 压力开关；5 压力开关按钮；6 塑料管；7 膜型听诊头；8 支架；9 膜片；10 谐振腔；11 弹簧）

该装置作为检测心音的探头，后面采用集成化的低噪声加法放大电路放大、去噪，再经声卡输入口与计算机连接，在屏幕上显示波形。该检测系统的放大倍数为 10～1000 倍自动调整，过载能力为 50 倍，采样频率在 5kHz～12kHz 可调，频率响应是 0.1～1300Hz。由于低频端处于人耳听阈以外，所以只有用心音检测系统才能真实地显示心音的波形特性。在日常的环境中，受检者可以隔着一件毛衣和衬衣进行检测，心声传感器一般放在二尖瓣听诊区附近，当选好听诊区，用力按下压力开关按钮后，检测装置才开始工作，能有效减少不必要的干扰，使受检者可以在比较宽松的条件下进行检测。

前置放大电路也是检测系统的重要单元，它具有高输入阻抗和低输出阻抗特性，以满足阻抗变换要求，有效放大信号，并具有温漂小、共模抑制比高等特点。根据心音信号的特点及其检测技术的要求，放大电路选用 OP07 高精度运放和 AD620 仪表放大器。

AD620 精度高，最大线性误差为 40ppm；最大失调电压为 50μV，失调漂移最大值为 0.6μV/℃。AD620 具有低噪声、低输入偏置电流和低功耗等特点，AD620 是一种低价格、高精度的仪表放大器，其增益由引脚 1 和 8 之间的阻抗决定，它仅需一个外接电阻便可达到共模抑制比 130dB 左右、增益 1000 倍的放大器，非常适合于医疗仪器使用。虽然理论上一片 AD620 的运放就可以满足放大器的指标要求，但考虑到调节的便利性及 AD620 可能产生 2mV 零点漂移，应在其前面增加一级 OP07 作为跟随器，补偿 AD620 的漂移，放大电路原理如图 7-2 所示。

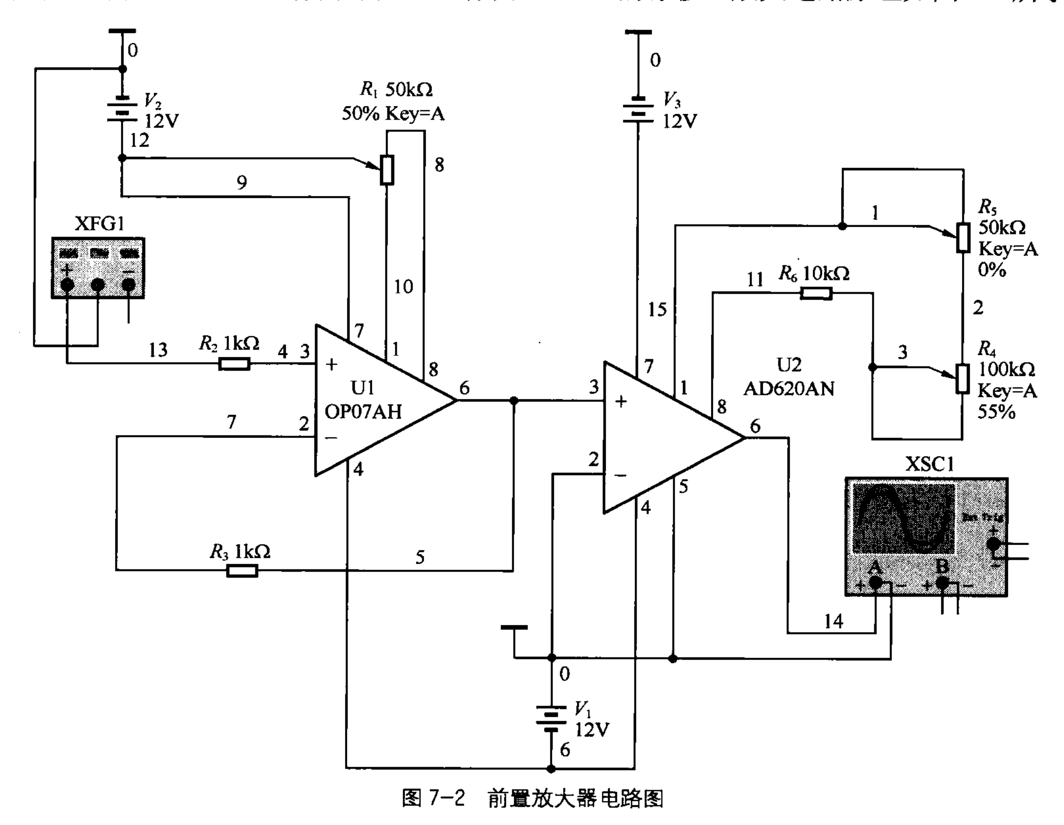


图 7-2 前置放大器电路图

其放大倍数为 $K=\dfrac{49.8\times10^{3}}{R_4+R_5+R_6}+1$。

R_1、R_4、R_5 为可调节电阻，R_1 用于调节零点。OP07 与低温度系数电阻 R_2 和 R_3 组成跟

随器。为避免程控开关引入的噪音信号，本放大器采用两级电位器进行增益调节，分别进行粗调和微调，实现增益的连续可调。R_4、R_5、R_6与 AD620 组成信号放大器，R_4采用 100kΩ可变电阻，用于粗调放大倍数；R_5为 10kΩ可变电阻，用于精调放大倍数，R_6的作用是对放大器起保护作用。

7.1.3 测量放大器

通常被检测的物理量通过传感器变换成模拟电信号，往往是很微弱的微伏级信号，需要通过放大器加以放大。现在市场可以采购到各种放大器（如通用运算放大器、测量放大器等），由于通用运算放大器一般都具有毫伏级的失调电压和每度数微伏的温漂，因此通用运算放大器一般不能直接用于放大微弱信号，而测量放大器则能较好地实现此功能。

测量放大器是一种带有精密差动电压增益的器件，由于它具有高输入阻抗、低输出阻抗、强抗共模干扰能力、低温漂、低失调电压和高稳定增益等特点，使其在检测微弱信号的系统中被广泛用作前置放大器。

测量放大器的电路原理图如图 7-3 所示。由图可见，测量放大器是由 3 个运放构成，并分为二级：第一级是两个同相放大器 A_1、A_2，因此输入阻抗高，第二级是普通的差分放大器，把双端输入变为对地的单端输出。下面以图 7-3 所示的测量放大器电路原理为例，讨论两个问题：测量放大器的增益和抗共模干扰能力。

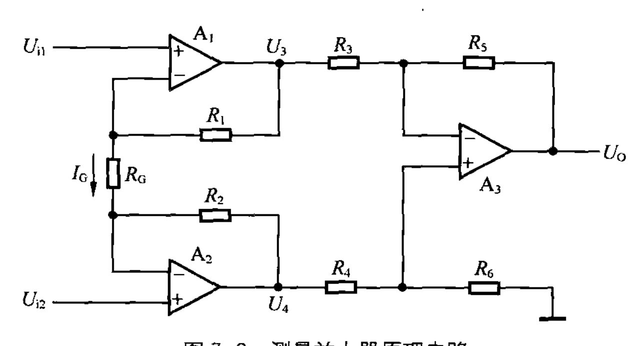


图 7-3 测量放大器原理电路

1. 测量放大器的增益

测量放大器的增益可用以下公式确定

$$K=\frac{U_O}{U_{i1}-U_{i2}}=\frac{(U_3-U_4)U_O}{(U_{i1}-U_{i2})(U_3-U_4)} \tag{7-1}$$

因为

$$U_3=U_{i1}+I_G R_1 \tag{7-2}$$

$$U_4=U_{i2}-I_G R_2 \tag{7-3}$$

$$I_G=\frac{U_{i1}-U_{i2}}{R_G} \tag{7-4}$$

所以

$$\frac{U_3-U_4}{U_{i1}-U_{i2}}=\frac{R_G+R_1+R_2}{R_G}$$

而测量放大器输出电压为

$$U_O = U_4\left[\frac{R_6}{R_4+R_6}\left(1+\frac{R_5}{R_3}\right)\right] - U_3\frac{R_5}{R_3} \tag{7-5}$$

为提高共模抑制比和降低温漂影响，测量放大器采用对称结构，即取 $R_1=R_2$，$R_3=R_4$，$R_5=R_6$，联立解式（7-1）～式（7-5），并整理得

$$K = \frac{U_O}{U_{i1}-U_{i2}} = -\left(1+\frac{2R_1}{R_G}\right)\frac{R_5}{R_3} \tag{7-6}$$

所以，通过调节外接电阻 R_G 的大小可以很方便地改变测量放大器的增益。

2．抗共模干扰能力

由图 7-3 可知，对于直流共模信号，由于 $I_G=0$，当 $R_3=R_4=R_5=R_6$ 时，$U_O=0$，所以测量放大器对直流共模信号的抑制比为无穷大。对于交流共模信号，情况就不一样了，因为输入信号的传输线存在线阻 R_{i1}、R_{i2} 和分布电容 C_1、C_2，如图 7-4 所示。显然，$R_{i1}C_1$ 和 $R_{i2}C_2$ 可分别对地构成回路，当 $R_{i1}C_1 \neq R_{i2}C_2$ 时，交流共模信号在两运放输入端产生分压，其电压分别为 U_{i1} 和 U_{i2}，且 $U_{i1} \neq U_{i2}$，所以 $I_G \neq 0$，对输入信号产生干扰。

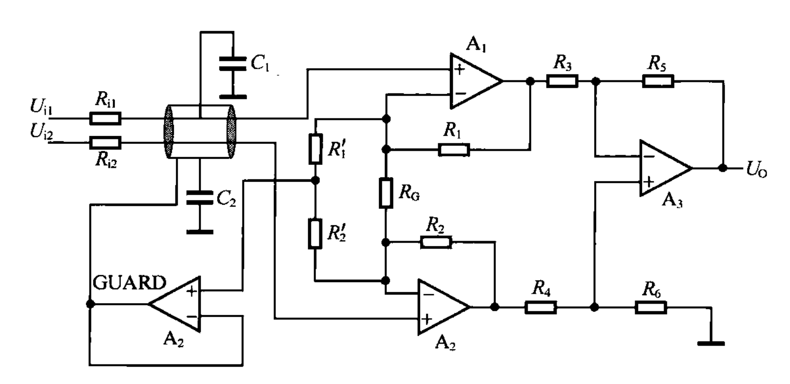


图 7-4　交流共模干扰影响及抑制方法

要抑制交流共模信号的干扰，可在其输入端加接一个输入保护电路和把信号线屏蔽起来，这就是所谓的“输入保护”。当 $R_1'=R_2'$ 时，由于屏蔽层和信号线间对交流共模信号是等电位的，因此 C_1 和 C_2 的分压作用就不存在，从而大大降低了共模交流信号的影响（因为正常使用的情况下，C_1 远远大于 C_2）。

虽然目前市场也有高精、低漂移的运算放大器，但在弱信号、强干扰的环境中应用，仍代替不了测量放大器，主要原因如下。

（1）为了提高抗共模干扰能力和抑制温漂影响，通常要求运放的两个输入电阻对称。这样，一则运放的输入阻抗受反馈电阻影响不可能做到很高，因此不适于作为多点检测的前置放大器（因为信号源内阻不同，放大器增益也不同）；二则调节增益不方便，因为要保证两输入端电阻对称，必须在改变反馈量（调节增益）的同时，也要相应调节另一输入端等效输入电阻。

（2）抗共模干扰的能力低于测量放大器，尤其是对交流共模信号，原因是它无法接入“输入保护”电路。

通过 multism 软件的仿真电路如图 7-5 所示。

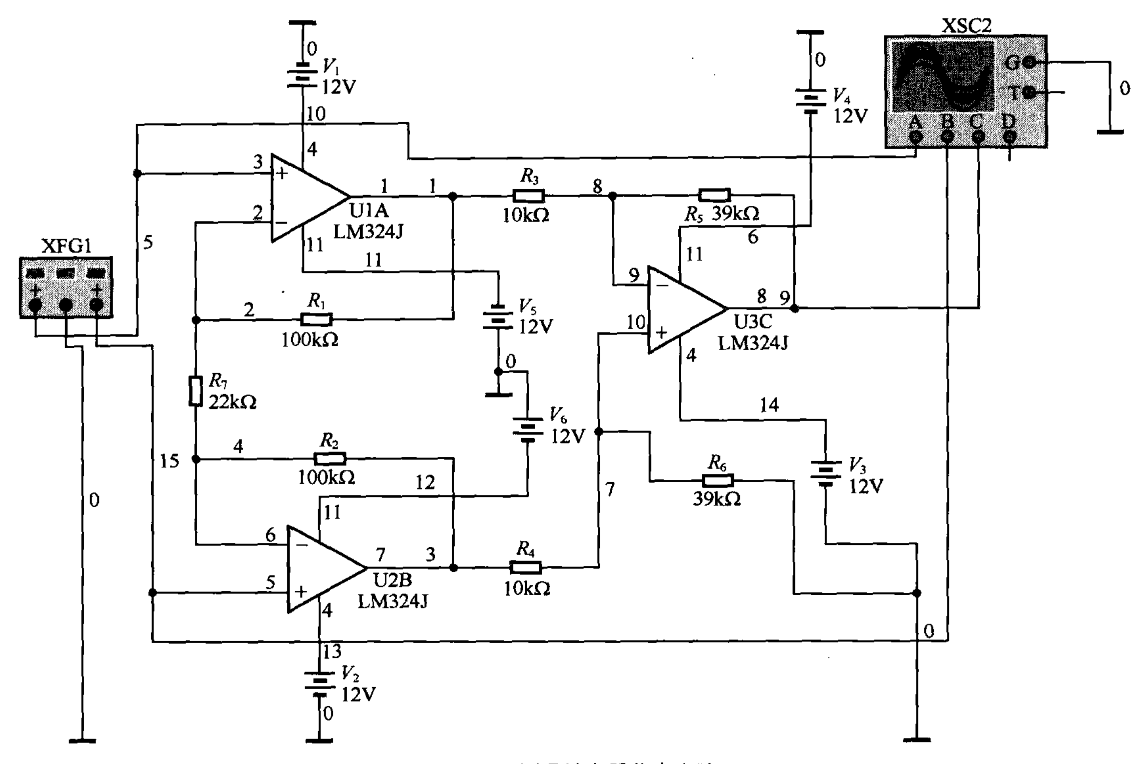


图 7-5　测量放大器仿真电路

该电路的电压放大倍数 K=39.35，但是，通过调节外接电阻 R_7 的大小可以很方便地改变测量放大器的增益。

7.1.4　多路数据采集系统

数据采集系统从严格的意义上来说，应该是用计算机控制的多路数据自动检测或巡回检测，并且能够对数据实行存储、处理、分析计算以及从检测的数据中提取可用的信息，供显示、记录、打印或描绘的系统。

数据采集系统一般由数据输入通道、数据存储与管理、数据处理、数据输出及显示这 5 个部分组成。输入通道要实现对被测对象的检测、采样和信号转换等工作；数据存储与管理要用存储器把采集到的数据存储起来，建立相应的数据库，并进行管理和调用；数据处理就是从采集到的原始数据中删除有关干扰噪声、无关信息，提取出反映被测对象特征的重要信息，另外，就是对数据进行统计分析，以便于检索，或者把数据恢复成原来物理量的形式，以可输出的形态输出到设备上，例如打印、显示、绘图等；数据输出及显示就是把数据以适当的形式进行输出和显示。

计算机技术的发展和普及提升了数据采集系统的技术水平。在生产过程中，应用这一系列可对生产现场的工艺参数进行采集、监视和记录，为提高产品质量、降低成本提供信息和

手段。在科学研究中，应用数据采集系统可获得大量的动态信息，是研究瞬间物理过程的有力工具。总之，不论在哪个应用领域中，数据的采集与处理越及时，工作效率就越高，取得的经济效益就越大。

随着"物联网"时代的临近，万物之间的距离似乎又被拉近了很多，万物都可以上网，"无处不在的计算"，更加方便快捷的"感知中国"，"感知世界"。这给传感器网络提出了更高的要求，需要数据采集能够更方便，精度更高，速度也更快。这里将数据采集分为有线数据采集和无线数据采集两类。

1. 有线数据采集

通过设计一数字电压表来简单介绍有线数据采集，本设计是在现场物理信号通过传感器采集，放大到 0～5V 的基础上，由单片机进行控制采集通道采集数据，并通过 LCD 显示。系统最多显示 8 路数据，为了突出重点，本例只介绍了基于单片机的数据采集控制，传感器和放大电路并没有介绍。整体框图如图 7-6 所示。

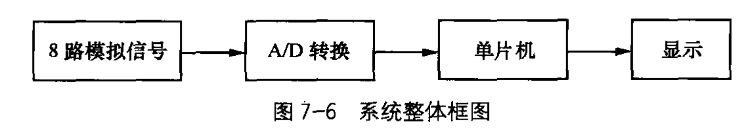


图 7-6 系统整体框图

ADC 将 0～5V 模拟信号转换为 00～FF 数字信号并传送给单片机，然后由单片机进行数据存储及数据处理，最终由显示器显示，完成对模拟信号的采集。Proteus 调试电路如图 7-7 所示。

模拟的电压信号直接从 ADC0809 的 IN0～IN7 输入，这里调试使用了 4 路信号。数字量由 OUT1～OUT8 输出并直接接到单片机 AT89S52 的 P0 端口（P0 端口作为输入口），P0 还作为 ADC 通道地址选通信号，由于 ADC0809 内部设有地址锁存器，所以通道地址由 P0 端口的第 3 位直接与 ADC0809 的 A、B、C 相连。通道基本地址为 0000H～0007H。将 P3.0 作为片选信号，在启动 A/D 转换时，由单片机 P3.0 控制 ADC 的启动转换。在读取转换结果时，用单片机 P3.1 与 OE 相连，用于打开三态输出锁存器。

由于 ADC0809 在进行 A/D 转换时需要有 CLK 信号，而此时 ADC0809 的 CLK 是接在 AT89S52 单片机的 P3.3 端口上，也就是要求从 P3.3 输出 CLK 信号供 ADC0809 使用，也就是说 CLK 信号用软件来控制的。

当 ADC0809 的 ALE 为高电平时，通道地址输入到地址锁存器中，下降沿将地址锁存，并译码。在 START 上升沿时，所有的内部寄存器清零，在下降沿时，开始进行 A/D 转换，此期间 START 应保持低电平。在 START 下降沿后 10μs 左右，转换结束信号变为低电平。EOC 为低电平时，表示正在转换，为高电平时，表示转换结束。OE 为低电平时，D0～D7 为高阻状态，OE 为高电平时，允许转换结果输出。

当 AT89S52 通过对 0000H～0007H（基本地址）中的某个端口地址进行一次写操作，即可启动相应通道的 A/D 转换。当转换结束后，ADC0809 的 EOC 端向 AT89S52 发出中断申请信号。AT89S52 通过对 0000H～0007H 中的某个端口地址进行一次读操作，即可得到转换结果。ADC0809 的参考电压 $V_{REF}=V_{CC}$，所以转换之后的数据要经过数据处理，在数码管上显示出电压值，实际显示的电压值为 D/256×VREF。

系统的程序框图如图 7-8 所示。

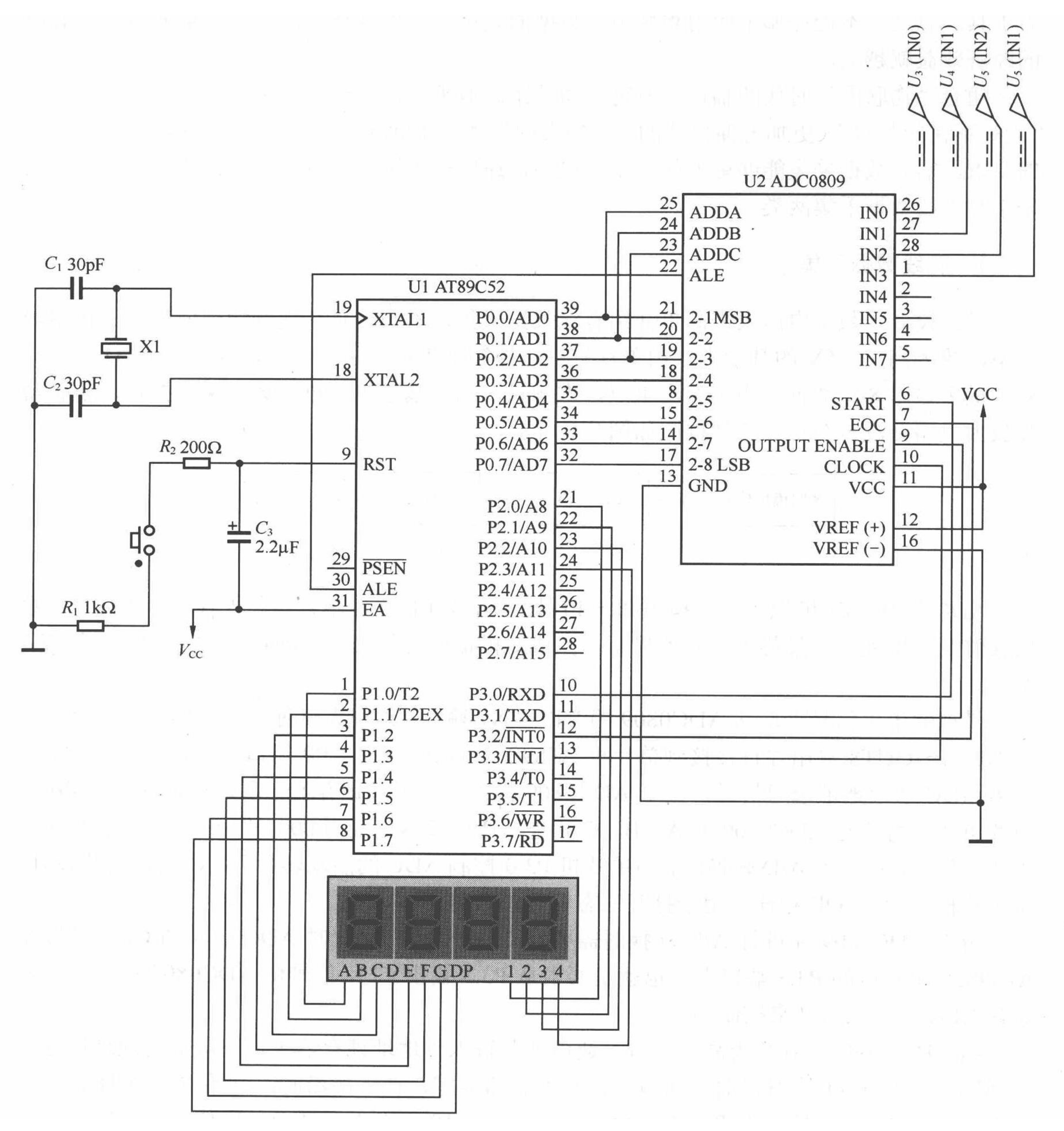


图 7-7　数字电压表调试电路

部分程序如下：

```
incude <AT89X52.H>
unsigned char code dispbitcode[]={0xfe,0xfd,0xfb,0xf7, 0xef,0xdf,0xbf,0x7f};
                                                          //显示的位码
unsigned char code dispbitcode[]={0x3f,0x06,0x5b,0x4f,0x66,0x7d,0x07,0x6f,0x00};
                                                          //显示的段码
unsigned char dispbuf[8]={10,10,10,10,0,0,0,0};           //显示的数据
```

```
unsigned char dispcount;
unsigned char getdata;
unsigned int temp;
unsigned char i;
sbit START=P3^0;
sbit OUTPUTENABLE=P3^1;
sbit EOC=P3^2;
sbit CLOCK=P3^3;
void main(void)
{
    START=0;
    OUTPUTENABLE=0;
    ET0=1;
    ET1=1;
    EA=1;
    TMOD=0x12;
    TH0=216;
    TL0=216;
    TH1=(65536-4000)/256;
    TL1=(65536-4000)%256;
    TR1=1;
    TR0=1;
    START=1;                    //START 上升沿时，所有的内部寄存器清零
    START=0;                    //START 下降沿时，开始进行 A/D 转换
    while(1)
    {
        if(EOC==1)              //EOC 为高电平时，表示转换结束
        {
            OUTPUTENABLE=1;     //OE 为高电平时，允许转换结果输出
            getdata=P0;
            OUTPUTENABLE=0;     //OE 为低电平时，D0～D7 为高阻状态
            temp=temp/128;
            i=5;
            dispbuf[0]=10;
            dispbuf[1]=10;
            dispbuf[2]=10;
            dispbuf[3]=10;
            dispbuf[4]=10;
            dispbuf[5]=0;
            dispbuf[6]=0;
            dispbuf[7]=0;
            while(temp/10)
            {
                dispbuf[i]=temp%10;
                temp=temp/10;
                i++;
            }
            dispbuf[i]=temp;
            START=1;
            START=0;
        }
```


图 7-8　系统程序框图

```
        }
    }
    void t0(void) interrupt 1 using 0
    {
        CLK=~CLK;
    }
    void t1(void) interrupt 3 using 0
    {
        TH1=(65536-4000)/256;
        TL1=(65536-4000)%256;
        P1=dispcode[dispbuf[dispcount]];
        P2=dispbitcode[dispcount];
        if(dispcount==7)
        {
            P1=P1|0X80;
        }
        dispcount++;
        if(dispcount==8)
        {
            dispcount=0;
        }
    }
```

2. 无线数据采集

目前，无线传感器的功能越来越强大，数据处理能力也越来越强，人们对无线通信功能的要求也越来越高。为此，各种无线设备进入人们的生活，在此基础之上，大量的无线设备业已进入工业生产领域。在工业或者工厂底层环境中，使用无线技术具有很多优势。利用无线技术可以解决工业生产中线路布局烦琐的困境，也可以避免恶劣环境对线路腐蚀等问题。

在此，把嵌入式技术与无线通信技术相结合，设计一种无线数据采集系统。该系统分为数据采集模块、数据传输/接收模块、数据处理模块，如图 7-9 所示。数据采集模块通过将传感器得到的模拟信号转换成数字信号，再通过无线芯片发送出去。A/D 转换芯片采用 MAX132，数据传输芯片采用无线传输/接收芯片 IA4421。用单片机配置 MAX132 和 IA4421 实现信号采集和发送。数据处理模块采用 S3C2410 芯片，并且配置无线芯片 IA4421 为无线接收模式。这样就实现了数据采集、传输与处理。

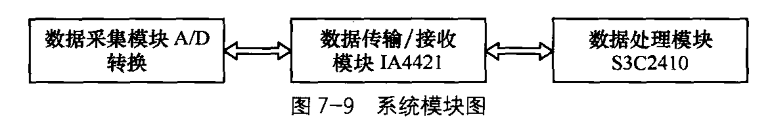


图 7-9 系统模块图

数据采集模块使用 A/D 转换芯片，通过把接收到的模拟信号转换成数字信号，再通过数据传输/接收模块将数字信号传输到数据处理模块。

MAX132 是 18 位外加 1 个符号位，具有高精度分辨率，以串行方式工作的 A/D 转换芯片。它可在–512～+512 mV 全范围内提供 2μV 的分辨率，精度可达±0.006%满量程。芯片较一般的积分型 ADC 具有更高的转换速度，可达每秒 100 次，简单的 4 线串行接口使其容易与其他所有的微处理器连接。MAX132 在普通工作方式下，典型供电电流为 60mA，在休眠模式下仅为 1μA。MAX132 还具有用于外部多路开关或可编程增益放大器的 4 个可编程的数

字输出，芯片内部还有可选 50Hz 工频的干扰抑制电路，芯片输入电流很小，仅为 10pA。MAX132 具有分辨率高、功耗低、价格低、体积小等特点，可广泛应用于远程数据采集、电池供电仪器仪表和传感器信号测量及工业过程控制等。

MAX132 的典型应用电路如图 7-10 所示。其中，元件参数是针对每秒转换 16 次，60 Hz 工频干扰抑制选择的。通过 2.5V 的高精度基准源，MAX872 分压后产生 545mV 的基准电压。待测电压信号从 INH1 和 INL0 端口差分输入，片选信号为 $\overline{CS}$，串行数据输入/输出端为 DIN、DOUT，通过时钟信号 Sclk 与单片机连接实现。

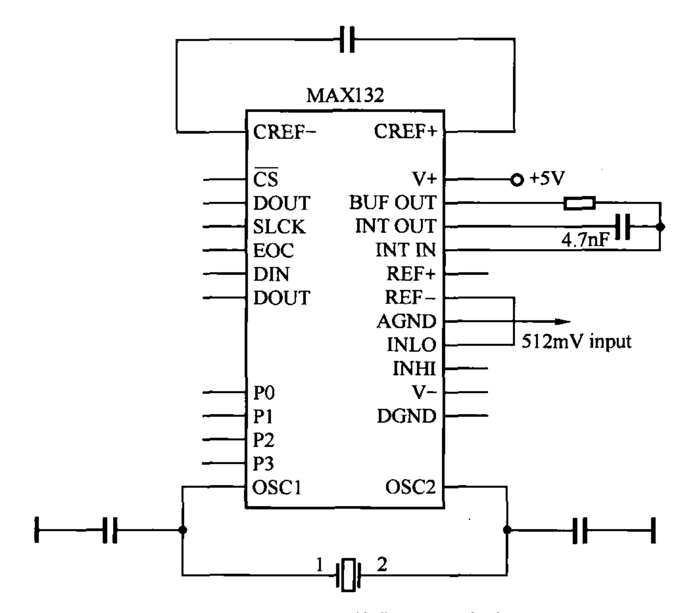


图 7-10 MAX132 的典型应用电路图

如图 7-11 所示，当 IA4421 芯片的 CLK/$\overline{CONV}$ 脚作为启动转换时，CLK/$\overline{CONV}$ 为高电平；若将 CLK/$\overline{CONV}$ 置低电平，则其 10ms 之内又变为高电平，这样就执行了一次转换，然后回到空闲模式。如果 CLK/$\overline{CONV}$ 一直处于低电平，就连续进行转换，直到 CLK/$\overline{CONV}$ 再次变为高电平。

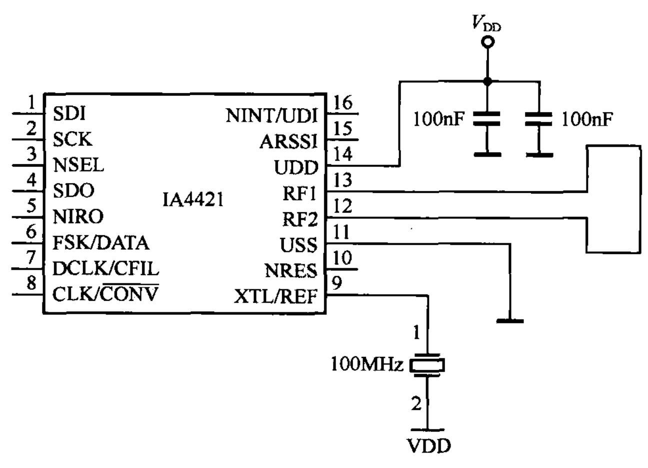


图 7-11 IA4421 电路接线图

数据传输/接收模块采用IA4421芯片。IA4421支持天线直接驱动，设计相当简单方便，并且通信距离长。IA4421是全集成的单晶片低功耗、多频道的FSK收发器，在无需申请注册的433MHz、868MHz、915MHz频段使用，设计完全符合FCC的ETSI认证，内部集成有高频功率放大器、低噪声放大器、I/Q转换混频器、基带滤波器、放大器、I/Q解调器等，所需的RF都已集成，只需一个晶振和几个去耦电容。

发射器的功能如下：

（1）发送采集数据指令，打开采集系统，为发送数据做好准备。

（2）与发射器实现自动对码。发射器控制软件程序流程图如图7-12所示，单片机上电初始化后，完成对IA4421的配制。此时，芯片被配置为发射模式，然后打开中断，等待数据输入，接收到采集的信号，最后将信息发送出去。

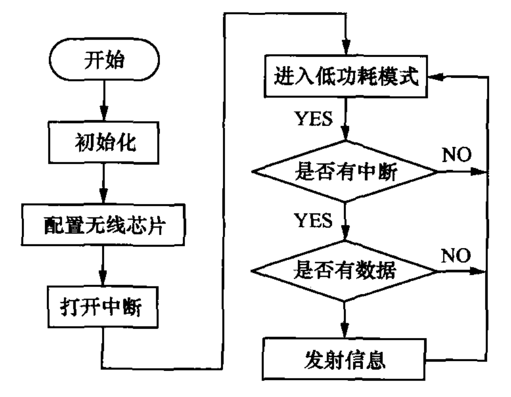


图7-12 发射器控制软件程序流程图

接收器的功能：

接收器主程序流程与发送器程序流程很相似，在此不再画流程图，这里仅对接收器作功能介绍：配置IA4421芯片为接收模式，发送准备好的信号。然后接收数据，再传输给处理单元，处理单元为S3C2410主芯片。

接收过程如下：

（1）配置无线接收芯片为接收模式，发送接收数据命令，通知IA4421准备好接收收据，且发送准备好的信号，通知发送模块发送数据。

（2）将接收到的数据进行运算分析，然后将结果反映到LCD上。应用程序具有分析功能，对信号的性能进行分析，且在信号出错的情况下做出报警处理。

7.2 信号输入电路

信号输入电路是将采集到的信号通过并行口或串行口输入到PC机上，输入方式可以是键盘、手写字符或者扫描仪等。本节简单介绍键盘输入和手写字符输入这两种输入方式的电路。

7.2.1 键盘输入电路

键盘由一组规则排列的按键组成，一个按键实际上是一个开关元件，是一种常用的输入设备。键盘通常包括有数字键（0～9），字母键（a～z，A～Z）以及一些功能键。操作人员可以通过键盘向计算机输入数据、地址、指令或其他控制符，实现简单的人机对话。

现在用的主要有独立式键盘和矩阵式键盘两种。独立式键盘只能用于键盘数量要求较少的场合，当按键数较多时，为了少占用I/O端口线，通常采用矩阵式键盘。矩阵式键盘的工作方式有3种，即编程扫描、定时扫描和中断扫描。下面简单介绍一种常用的矩阵式键盘电路。

CPU对键盘的扫描采取程序控制方式，一旦进入键盘扫描状态，则反复地扫描键盘，等待用户从键盘上输入命令或数据。而在执行键输入命令或处理输入数据过程中，CPU将不再响应输入要求，直到CPU重新扫描键盘为止。

下面使用 Mutilsim 软件设计一个 4×4 矩阵键盘的识别电路，如图 7-13 所示。采用 8051 单片机，40 脚接+5V 电源，20 脚接地。8051 的并行口 P1 接 4×4 矩阵键盘，以 P1.0～P1.3 为列线，P1.4～P1.7 为行线。P3 端口输出至共阳极七段显示数码管，数码管显示所按矩阵键盘的符号。程序框图如图 7-14 所示。

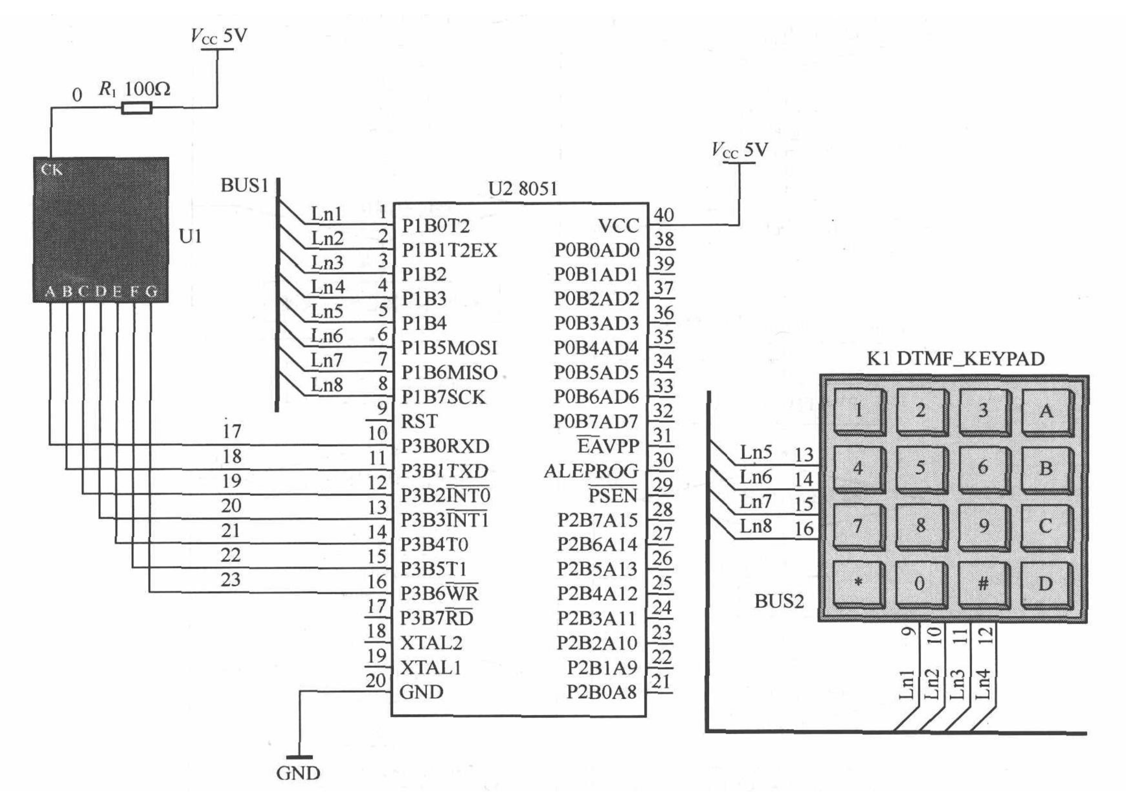


图 7-13　4×4 矩阵键盘识别电路

程序说明：P1 端口初始化全置 1，P1.4=0，扫描确定按键在第一行，若某列线为 0。则表示第一行对应列字符输入，并由 P3 端口输出其 7 段数码管显示。P1.5=0，扫描确定按键第二行符号，否则扫描线为 0，同理数码管输出所按键。

部分源程序如下：

```
#include"htc.h"
unsigned char code table[]=
{0x40,0x79,0x24,0x30,0x19,0x12,0x02,0x78,
0x00,0x10,0x08,0x03,0x46,0x21,0x06,0x0E};        //4×4 矩阵键盘按键值
unsigned char temp;
unsigned char key;
unsigned char i,j;
void main(void)
{
   while(1)
   {
      P1=0xff;
      P14=0;                                      //判断第一行是否有键按下
```

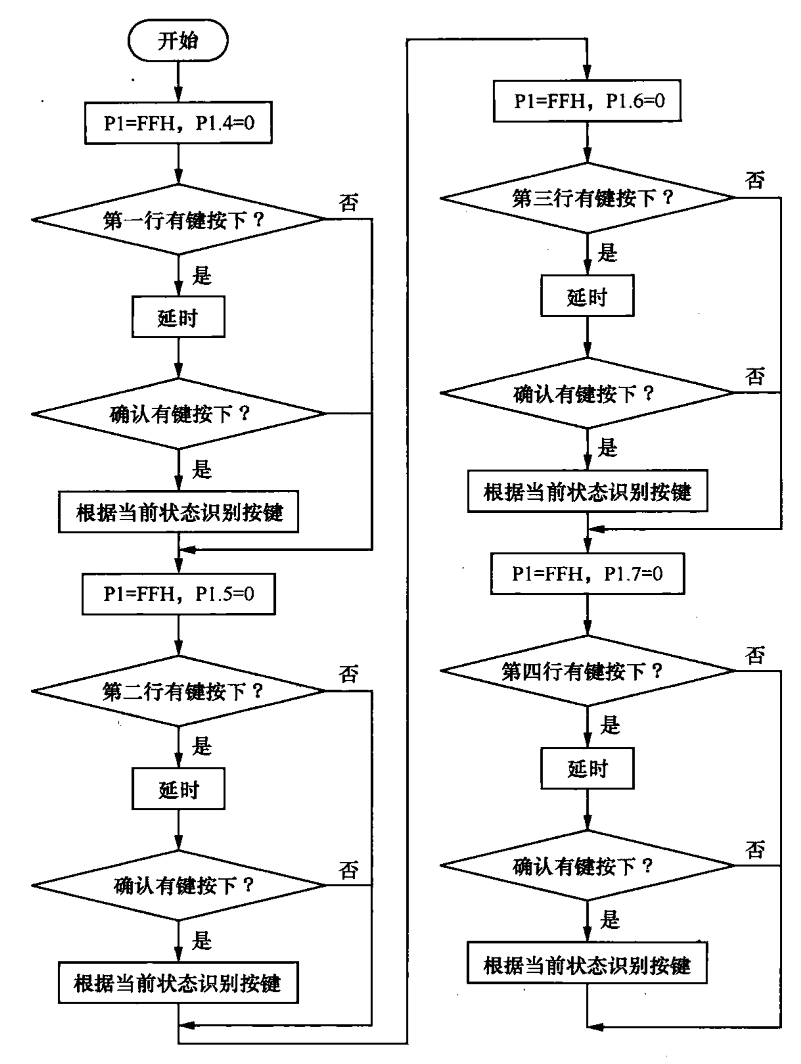


图 7-14　4×4 矩阵键盘识别程序框图

```
temp=P1;
temp=temp&0x0f;
if(temp!=0x0f)
{
    for(i=1;i>0;i--)
        for(j=1;j>0;j--)
        temp=P1;
        temp=temp&0x0f;
    if(temp!=0x0f)
    {
        temp=P1;
        temp=temp&0x0f;
        switch(temp)
        {
            case 0x0e: key=1; break;        //第一列按键“1”
            case 0x0d: key=2; break;        //第二列按键“2”
            case 0x0b: key=3; break;        //第三列按键“3”
```

```
                case 0x07: key=10; break;      //第四列按键 "A"
            }
                temp=P1;
            P3=table[key];
            temp=temp&0x0f;
            while(temp!=0x0f)
            {
            temp=P1;
            temp=temp&0x0f;
            }
        }
    }
  ...
  ...                                          //以下程序读者可以参考上面的加以完成
  }
}
```

7.2.2 手写字符输入电路

手写输入法是用手写笔在手写板上书写字符以输入字符的一种输入方法。本小节介绍一个手写字符输入电路，如图 7-15 所示。图中 2 片 CD4067（单 16 通道模拟开关）组成 16×16

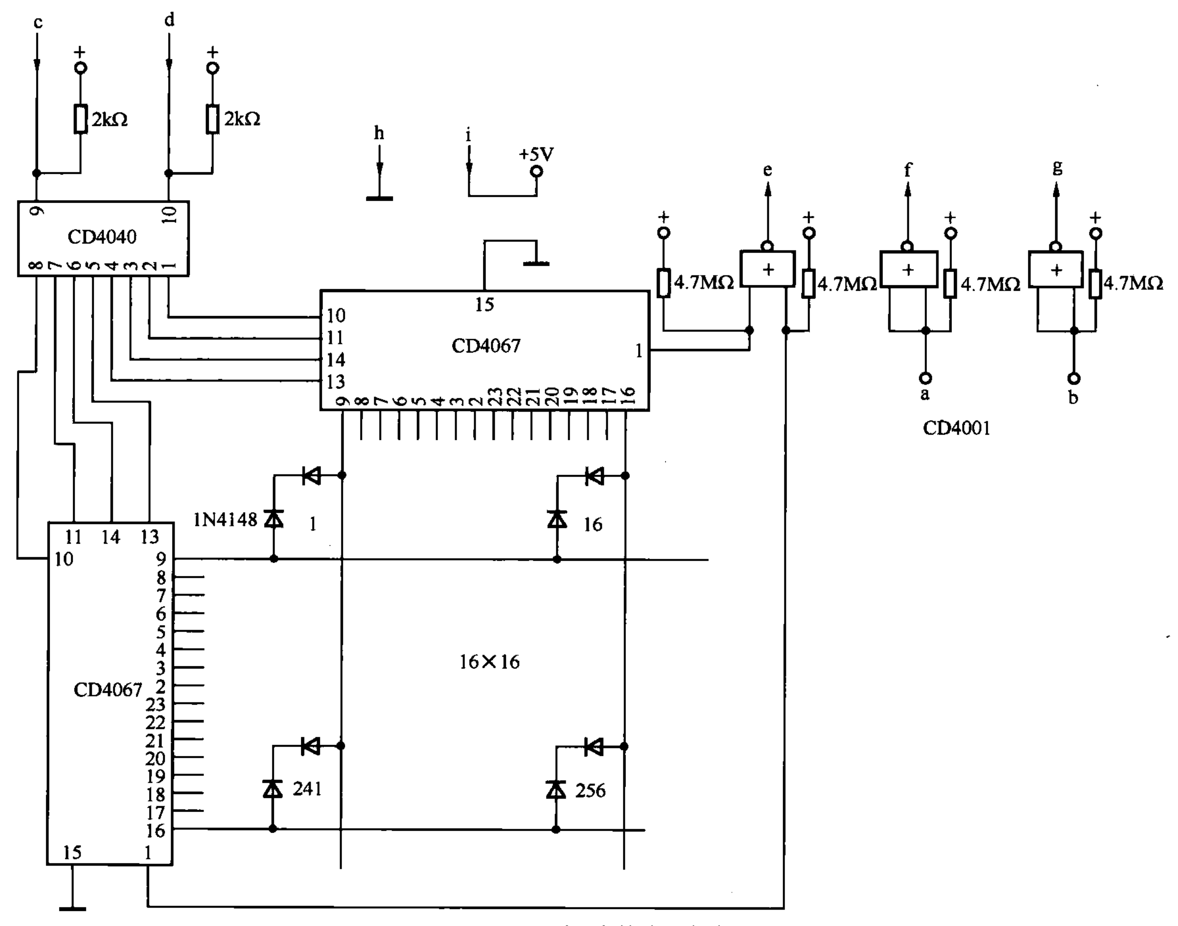


图 7-15 手写字符输入电路

点阵，由 1 片 CD4040（12 级二进制计数器驱动其从第 1 点扫描到第 256 点，2 片 CD4067 的公共端接在或非门的 2 个输入端。当扫描到点阵的某一点时，如果该点没有被接入低电平，则 2 片的 CD4067 的公共端无输出，或非门的输入端被 2 个 4.7MΩ电阻拉成高电平，其输出为低电平。如果该点被接入低电平，则 2 片 CD4067 的公共端均输出低电平，或非门输出为高电平。低电平的接入方法用了人体导电的原理，操作者右手手腕上戴一个自制的导电环，导电环与电路地线相连。当右手手指触摸到点阵的某一点，而 CD4067 又扫描到该点时，操作者手腕到手指之间的电阻与 4.7MΩ的电阻形成分压电路，由于人体的电阻远小于 4.7MΩ，所以或非门的输入为低电平，输出为高电平。点阵中的二级管是起隔离作用的，图中另外两个或非门各组成一个触摸钮，是当作功能键使用的。本电路共有 4 根输入线，3 根输出线与计算机相连。输入线中 2 根是电源正负级，1 根是清零线，1 根是扫描时钟输入线。输出线中 1 根是点阵输出线，2 根是功能键输出线。

7.3 信号显示电路

在应用系统或智能化仪器仪表中一般都配有显示器，以便人们及时掌握信息变化情况，并适时处理和应对这些变化。常用的显示器件有显示和记录仪表、CRT 显示终端、LED 数码显示器、LCD 显示器和大屏幕显示器。

显示记录仪表能连续显示和记录，但价格比较贵，而且它们的显示方式是模拟显示，读数不直观、不方便、不准确。这种显示器多数适用于企业的技术改造，在新设计的微型机自动化系统中不宜采用。CRT 显示终端直观、灵活，不但可以显示数字，而且还可以显示绘面及各种报表。目前 CRT 显示终端用于大、中型控制系统中，在小型控制系统和智能化仪器仪表中还是很少采用。

LED 数码显示管具有工作可靠、显示醒目、响应速度快、易于匹配、结构简单、体积小、功率低和寿命长等特点，被各种应用系统及智能化仪器仪表大量采用。LCD 显示器以其功耗极低、显示多样的显著特点，几乎遍布了所有的显示领域。从电子表到计算器，从袖珍仪表到人机界面，LCD 显示器几乎没有不涉及的电子产品。大屏幕显示器具有显示清晰、视觉范围宽等特点，主要用于车站、码头、体育场馆、大型生产装置的现场等。

本节就 LED 数码显示电路和 LCD 液晶显示电路作一个简单介绍。

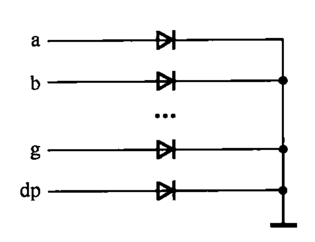
7.3.1 数码显示电路

LED 显示块是由发光二极管显示字段组成的显示器。按显示字符的形状来分，有七段和“米”字段两种；按接线方式分类，有共阳极和共阴极两种。七段 LED 显示块的结构及外形图如图 7-16 所示。

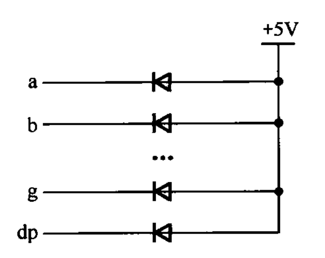
共阴极 LED 显示块的发光二极管的阴极连接在一起，通常将公共阴极接地。当某个发光二极管的阳极接高电平时，该发光二极管就会被点亮，相应的发光段就会发光显示。同样，共阳极 LED 显示块的发光二极管的阳极连接在一起。接线时，通常将公共阳极接在直流电源的正极。当某个发光二极管的阴极接低电平时，该发光二极管就会被点亮，相应的发光段就会发光显示。

注意：公共阴极接地（或共阳极接地）是要接限流电阻，避免因电流过大而把发光二极

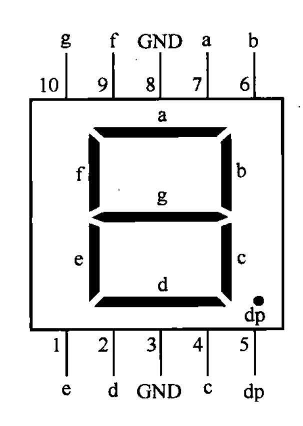
管烧掉。



(a) 共阴极 LED 字型码显示器接线原理



(b) 共阳极 LED 字型码显示器接线原理



(c) LED 字型码显示器外行图

图 7-16 七段 LED 显示块的结构及外形图

LED 显示块的引脚“dp”表示显示小数点，在七段 LED 显示块和“米”字段 LED 显示块中都有“dp”显示段。

LED 显示器的显示方式有两种：一种是静态显示；一种是动态显示。

在静态显示中，各位相互独立，每位 LED 需要一个译码器对该位进行译码。静态显示的最大优点是编程容易、管理简单，但这种显示方式接口占用硬件资源和连线资源比较多。

在显示位数较多的情况下，一般采用动态显示方式。该显示方式中所有显示数码管共用一个译码器。在某一个时刻，只让某一位的位选线处于选通状态，而其他各位的位选线都处于关闭状态，同时，段选线上输出相应位要显示字符的字型码。这样，在同一时刻 LED 显示中只有选通的那一位显示出字符，而其他各位则是熄灭的。同样道理，在下一时刻，只让下一位的位选线处于选通状态，而其他各位的位选线都处于关闭状态。同时，在段选线上输出相应位将要显示字符的字型码，则同一时刻，只有选通位可以显示出相应的字符，而其他各位则是熄灭的，如此循环下去，就可以使各位 LED 显示出将要显示的字符，只要达到视觉暂留的时间（25Hz 以上较为理想），人眼看上去就和静态显示相同的效果。这种 LED 显示方式就称为 LED 显示器的动态显示。

7.3.2 液晶显示电路

液晶是一种介于液体和固体之间的热力学的中间稳定相，在一定的温度范围内，它既有液体的流动性和连续性，又有晶体的各向异性。液晶显示器（Liquid Crystal Display，LCD），

是一种功耗极低，用途非常广泛的显示器件。实际应用中，一般采用两种办法使液晶显示器显示，一种是用集成电路芯片组装成的 LCD 控制器；另一种是直接采用已经将控制部分、RAM、ROM 和 LCD 连接在一起的液晶显示模块 LCM。

液晶显示的驱动方式是由电极引线的选择方式确定的，一般有静态驱动和“时分割”驱动两种。液晶显示器的驱动和 LED 驱动有很大不同。对于 LED 显示驱动，当在 LED 显示器的两端加上恒定的导通或截止电压后，便可以控制 LED 显示器的亮或灭。而对于 LCD 显示器，由于其电极的两端不能加恒定的直流电压，因此给驱动带来了一定的复杂性。驱动 LCD 显示器时，一般应在它的背极（公共极）加上恒定的交变方波信号，通过控制前级的电压变化，而在 LCD 的两级间产生所需的零电压或两倍幅值的交变电压，以达到 LCD 亮或灭的控制。

利用 PIC 单片机驱动 LCD 显示屏显示预定字符的电路如图 7-17 所示。PIC 单片机 14 脚接+5V 电源，5 脚接 0V 电源。LCD 液晶显示屏电源端接 5V 电压，液晶驱动电压接 5V 电压。PIC 分别通过 RA0 连接 LCD 的串行的数据口（并行的读写选择信号口）RW；RA1 连接 LCD 的串行的片选信号口（并行的指令/数据选择信号口）RS；RA2 连接 LCD 的并行的使能信号（串行的同步时钟）E 来控制 LCD 的显示。而用 RB 口向 LCD 写数据，RBOINT、RB1～RB7 分别连接 LCD 的 D0～D7。

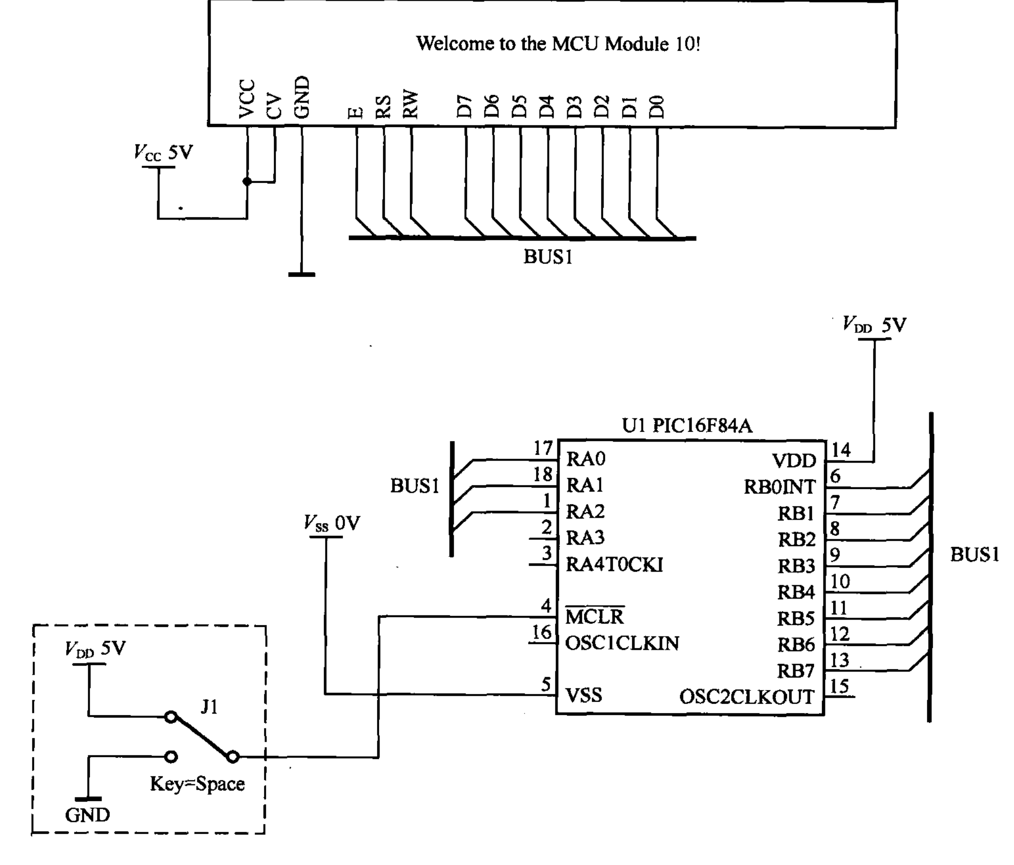


图 7-17　LCD 显示器控制电路

Multisim10 软件中单击“MCU/MCU PIC16F84A U1/MCU Code Manager”菜单命令，进

入微控制器代码管理器对话框，可以进行程序的编译，编译完成后单击“MCU/MCU PIC16F84A U1/Debug View”菜单命令进行调试，如果程序有错误，单击出错提示信息，光标会自动跳到程序出错处。检查错误并修改，直到编译通过。源程序通过后，单击启动仿真按钮，则可进行加载仿真。

注：详细的源程序请参考 NI Multisim 10 评估版中“Samples/MCU Sample Circuit”文件夹中的 LCD_Display.asm。

单击“MCU/MCU PIC16F84A U1/Memory View”菜单命令，可以观察到存储器的内部数据。

7.4 信号转换电路

从信息形态变化的观点可以将各种转换分为 3 种：(1) 从自然界物理量到电量的转换（传感器）；(2) 电量之间的转换（转换电路）；(3) 从电量到物理量的转换。

对信号的处理和分析往往应用数字电路或计算机进行处理，在处理之前，一般先将模拟信号转化为数字信号，数字电路或计算机对数字信号处理完毕后，还要把数字信号再转化为模拟信号，作为电路的输出。把模拟信号转换成数字信号的过程称为模/数转换，简称 A/D (Analog to Digital)；把数字信号转换为模拟信号的过程称为数/模转换，简称 D/A (Digital to Analog)。实现 A/D 转换的电路一般由 A/D 转换器完成，简写为 ADC (Analog to Digital Converter)；实现 D/A 转换的电路一般由 D/A 转换器完成，简写为 DAC (Digital to Analog Converter)。

目前，许多芯片公司都推出了单芯片的 DAC 或 ADC，也有许多 MCU 上集成了 DAC 或 ADC，随着集成电路技术的发展，其转换精度和转换速度等技术指标也越来越高。

7.4.1 数模 D/A 转换电路

数字量是用代码按数位组合起来表示的，对于有权码，每位代码都有一定的权。为了将数字量转换成模拟量，必须将每 1 位代码按其权的大小转换成相应的模拟量，然后将这些模拟量相加，即可得到与数字量成正比的总模拟量，从而实现了数模转换。

n 位 D/A 转换器的方框图如图 7-18 所示。

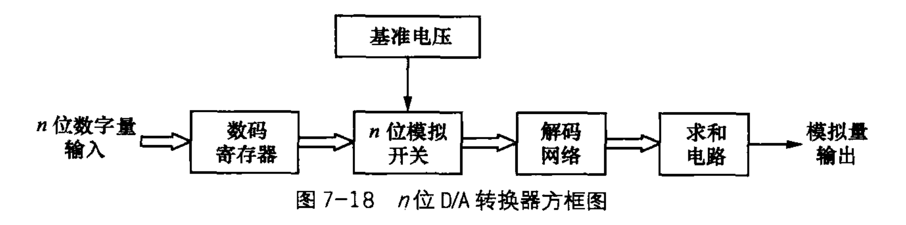


图 7-18　*n* 位 D/A 转换器方框图

D/A 转换器由数码寄存器、模拟电子开关电路、解码网络、求和电路及基准电压几部分组成。数字量以串行或并行方式输入并存储于数码寄存器中，寄存器输出的每位数码驱动对应数位上的电子开关，将在电阻解码网络中获得的相应数位权值送入求和电路。求和电路将各位权值相加便得到与数字量对应的模拟量。

D/A 转换器按解码网络结构不同分为 T 形电阻网络、倒 T 形电阻网络 D/A 转换器、权电

流 D/A 转换器及权电阻网络 D/A 转换器等。按模拟电子开关电路的不同，D/A 转换器又可分为 CMOS 开关型和双极型开关 D/A 转换器。其中双极型开关 D/A 转换器又分为电流开关型和 ECL 电流开关型两种，在速度要求不高的情况可选用 CMOS 开关型 D/A 转换器。如要求高的转换速度则应选用双极型电流开关 D/A 转换器或转换速度更高的 ECL 电流开关型 D/A 转换器。

倒 T 形电阻网络 D/A 转换器具有较高的转换速度，但由于电路中存在模拟开关电压降，当流过各支路的电流稍有变化时，就会产生转换误差。为进一步提高 D/A 转换器的精度，可采用权电流 D/A 转换器，4 位权电流 D/A 转换器原理电路如图 7-19 所示。

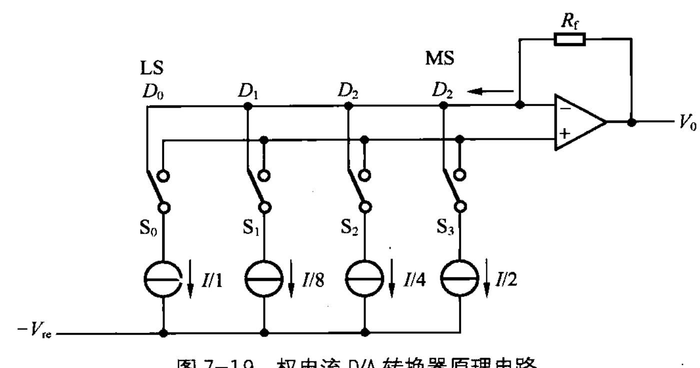


图 7-19　权电流 D/A 转换器原理电路

当输入数字量的某一位代码 D_i=1 时，开关 S_i 接运算放大器的反相端，相应权电流输入求和电路；当 D_i=0 时，开关 S_i 接地。分析该电路，可得出

$$v_0 = i_\Sigma R_f = R_f\left(\frac{I}{2}D_3 + \frac{I}{4}D_2 + \frac{I}{8}D_1 + \frac{I}{16}D_0\right)$$

$$= \frac{I}{2^4} \cdot R_f (D_3 \cdot 2^3 + D_2 \cdot 2^2 + D_1 \cdot 2^1 + D_0 \cdot 2^0)$$

$$= \frac{I}{2^4} \cdot R_f \sum_{i=0}^{3} D_i \cdot 2^i \qquad (7\text{-}7)$$

采用了恒流源电路后，各支路电流的大小均不受开关导通电阻和压降的影响，这就降低了对开关电路的要求，提高了转换精度。如将图 7-19 中所示恒流源采用具有电流负反馈的 BJT 恒流源电路，即可得实际的权电流 D/A 转换器电路。

为验证 D/A 转换器的功能，可以在 Multisim10 元器件库中的 Mixed 组中选择 D/A 转换器 DAC，D/A 转换器有两种类型：一种是电流型 DAC，即 IDAC；另一种是电压型 DAC，即 VDAC。以 VDAC 为例设计一个电路，如图 7-20 所示。按下仿真开关按钮，打开示波器，可以看到波形为 1kHz 的 16 阶梯锯齿波。

7.4.2　模数 A/D 转换电路

为了将时间连续、幅值也连续的模拟量转换为时间离散、幅值也离散的数字信号，A/D 转换一般要经过取样、保持、量化及编码 4 个过程。在实际电路中，这些过程有的是合并进行的，如取样和保持，量化和编码往往都是在转换过程中同时实现。

A/D 转换器的种类很多，按其工作原理的不同分为直接 A/D 转换器和间接 A/D 转换器两

类。直接A/D转换器可将模拟信号直接转换为数字信号，这类A/D转换器具有较快的转换速度，其典型电路有并行比较型A/D转换器、逐次比较型A/D转换器。而间接A/D转换器则是先将模拟信号转换成某一中间量（时间或频率），然后再将中间量转换为数字量输出。此类A/D转换器的速度较慢，典型电路时双积分型A/D转换器、电压频率转换型A/D转换器。常用的集成逐次比较型A/D转换器有ADC0808/0809系列（8位）、AD575（10位）、AD574A（12位）等。

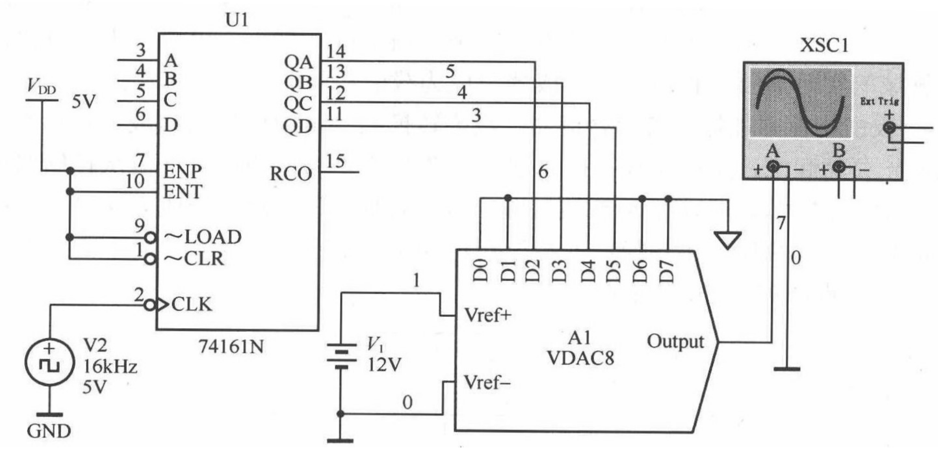


图7-20 D/A转换器电路

为验证A/D转换器的功能，可以在Multisim10元器件库中的Mixed组中选择A/D转换器ADC。ADC的主要功能是将输入的模拟信号转换成8位的数字信号输出。

基于ADC的A/D转换器仿真电路如图7-21所示，在电路中采用滑动变阻器构成一个分压电路。将滑动端接至ADC的模拟信号输入端，通过改变滑动变阻器R_1的大小，即可改变输入模拟信号的大小。ADC输出的高4位和低4位分别接一个数码管，显示输入模拟信号的转换结果。可以看到输入模拟量通过ADC转换后在数码管上显示出相对应的数字量。本章最后一节的范例温度测量仪中的A/D转换可以参考此电路设计完成。

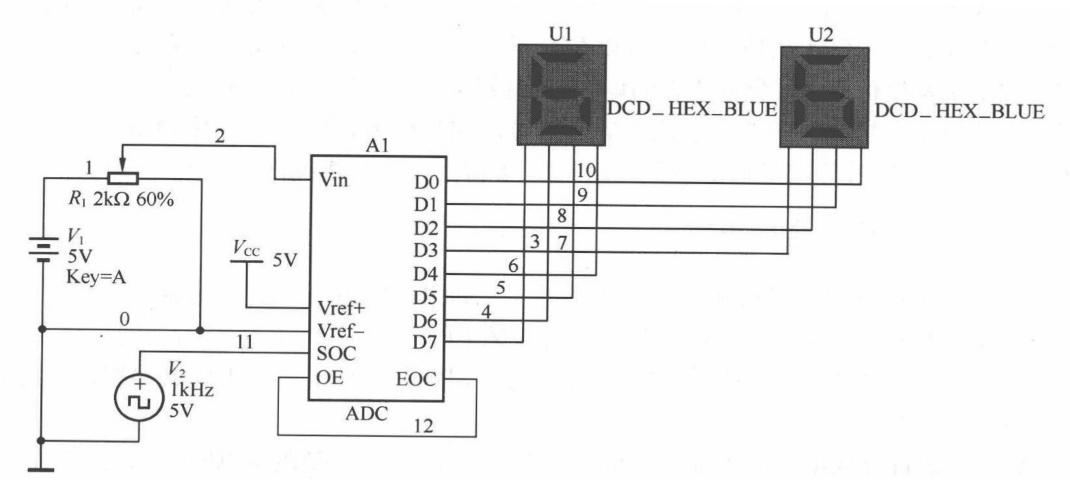


图7-21 A/D转换器电路

7.5 信号合成电路

7.5.1 直接数字合成器

直接数字合成（DDS）技术 1971 年首次由 Tierney 提出，相对传统频率合成电路具有频率转换时间短、频率分辨率高、便于集成、可靠性高、方便调制等优势。但受到当时微电子技术和数字信号处理技术的限制，未能有效应用。由于近年来大规模数字集成电路技术的发展和电子领域的实际需要，直接数字合成器已成为载波信号源的第一选择。

DDS 一般由相位累加器、波形存储器、数模转换器及低通滤波器组成，其基本原理就是将波形数据先存储起来，然后在频率控制字 K 的作用下，通过相位累加器从存储器中读出波形数据，最后经过数/模转换和低通滤波后输出频率合成。这种频率合成方法可以获得高精度频率和相位分辨率、快速频率转换和低相位噪声的频率信号，而且结构简单集成度高。

DDS 的原理框图如图 7-22 所示。

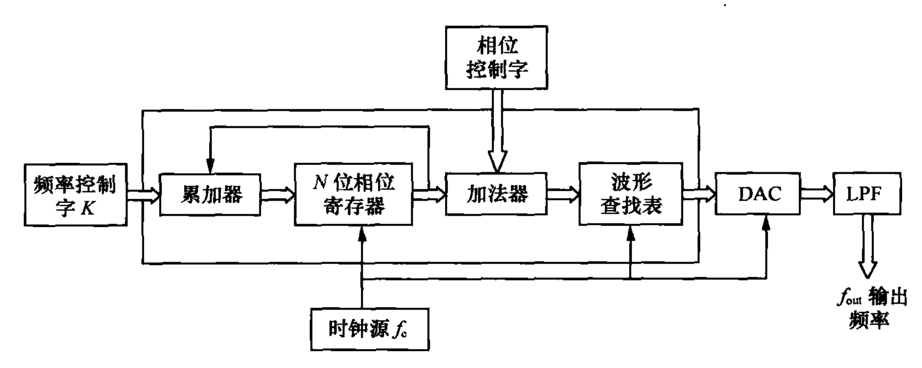


图 7-22 DDS 原理框图

频率控制字 K 和相位控制字 P 分别控制 DDS 输出波形的频率和相位。DDS 系统的核心是相位累加器，它由一个累加器和一个 N 位相位寄存器组成。每来一个时钟脉冲，相位寄存器以步长 K 增加。相位寄存器的输出与相位控制字相加，其结果作为波形查找表的地址。

波形查找表由 ROM 构成，内部可以存有正弦波、方波、锯齿波和三角波的一个完整周期波形的数字幅度信息，每个查找表的地址对应波形中 0°～360° 范围内的一个相位点。查找表把输入的地址信息映射成波形的数字幅度信号，同时输出到数模转换器 DAC 的输入端，DAC 输出的模拟信号经过低通滤波器（LPF），可得到一个频谱纯净的波形。

7.5.2 人工语音合成电路

语音是人类最有效，最快捷的信息传递形式。如果能够用语音交互功能来实现人与计算机之间的信息传递，此时的计算机才具有真正意义上的人工智能。在语音信号处理研究领域，计算机可以感知人类的发音所形成的语音称为语音识别，计算机可以生成需要的传递内容的语音称为语音合成。

HY 系列语音合成电路的内部原理框图如图 7-23 所示。控制逻辑电路、地址计数器、振荡器和 D/A 转换器等，是所有语音合成电路必有的相同的电路单元。所不同的仅是语音存储

器（ROM）中的客户专用语音信息资料。HY 系列语音合成电路有下列基本特征：

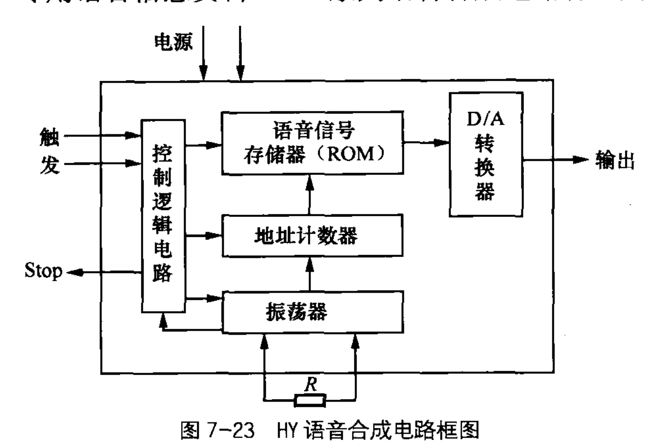


图 7-23 HY 语音合成电路框图

（1）单电源供电，工作电压为 2.4～6V。

（2）静态工作电流小于 2μA。

（3）语音长度为 3～48s，取样频率最高可达 22kHz。

（4）可分段触发输出不同语音段，且每段长度可不相同。可设置多个触发端，分别控制多段语音输出。

（5）可由开关、按钮或光敏电阻等启动，触发方式分为电平触发和脉冲触发两种。电平触发是指将触发电平保持在触发端，使语音电路连续输出语音，电平中断后，语音输出完毕即停止。脉冲触发是指触发端电平（一般正跳变）一次，语音电路输出语音一次后即自动停止，而电平的保持不能使语音电路连续工作。

7.5.3 功率放大器

功率放大器（简称功放）的作用是给负载 R_L 提供一定的输出功率。功率放大器的常见电路形式有 OTL（Output Transformerless）电路和 OCL（Output Capacitorless）电路。有用集成运算放大器（简称运放）和晶体管组成的功率放大器，也有专用集成电路功率放大器。

一般对功放的要求有：（1）根据负载要求提供所需要的输出功率；（2）效率要高；（3）非线性失真要小；（4）带动负载能力强。根据这些要求，一般多选用工作在甲乙类的射级输出器构成互补对称功率放大电路。这里介绍一种单电源功率放大电路，如图 7-24 所示。

首先调节静态工作点，函数发生器 XFG1 断开，输入端对地短路。单击仿真开关按钮，K_1 闭合，K_2 断开，用万用表测量 A 点的电位，调节电位器 R_7 的大小，使得 $U_A=V_1/2=3V$。

图中开关 K_1 的作用是电路是否加自举，当闭合时，加自举，断开时，不加自举。

将示波器接在电路的输出端和输入端，单击“仿真开关”按钮进行仿真，调节 XFG1 的输入信号幅度，观测示波器 XSC1 上输出电压的波形。逐渐增大输入电压的幅值，当用示波器观察到输出电压波形为临界削波时，减小输入电压幅值，使输出电压波形失真刚好消失，这时输出电压为电路的最大输出电压。

K_2 开关控制是否加入二极管 VD_1、VD_2，分别通过打开或关闭 K_2 观察示波器波形，可见加入二极管 VD_1、VD_2 后，能够使三极管 VT_2、VT_3 处于微导通状态，进而使功率放大电路的输出交越失真减小。

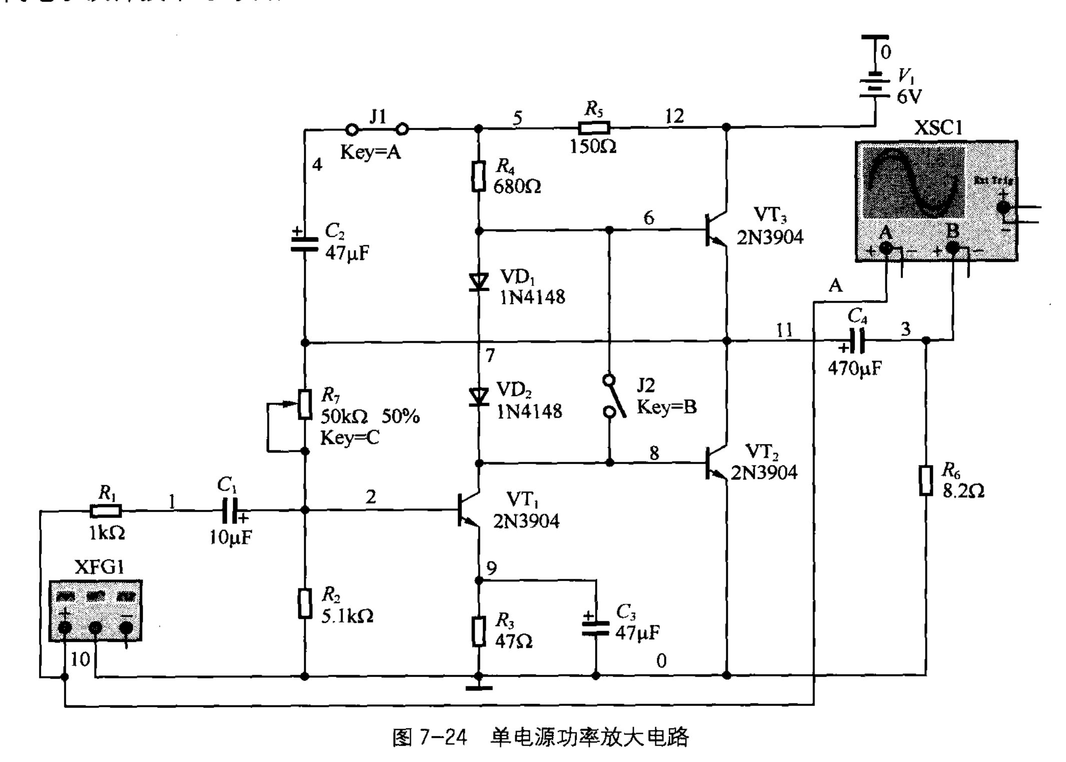


图 7-24　单电源功率放大电路

注意：不能让二极管 VD_1、VD_2 支路断开，否则 VT_2、VT_3 将过流烧毁。

7.6 信号分解电路

7.6.1 同步数字信号复用分解电路

同步信号复用电路包括：帧同步延迟电路、复用地址产生电路、复用串并转换电路、多路选择电路。同步信号分解电路包括：分解地址产生电路和分解串并转换电路。如图 7-25 所示。

复用地址产生电路输出复用读地址信号、复用写地址信号和选择使能信号；复用串并转换电路输出多路并行信号，它由多个 RAM 存储器实现多路同步数字信号的缓存，多路选择电路实现单路串行同步数字信号的组帧；多路选择电路输出复用后的单路数字同步信号。分解地址产生电路输出分解读地址信号、分解写地址信号和分解使能信号；分解串并转换电路由 RAM 存储器实现单路同步数字信号的多字节的缓存，并分解为多路并行同步数字信号。该复用分解电路用 RAM 存储器实现了字节的缓存，可节省大量的触发器，具有电路设计简单、经济实用等特点。

7.6.2 FPGA 分频器电路

在实际的电路设计中，通常使用一个时钟信号源，经分频得到所需的频率。对于整数分频的实现较为简单，通常由计数器或计数器的级联构成。但对半整数分频的实现较为困难，设计的思想是：设计一个模 N 计数器，再设计一个脉冲扣除电路，每来 $N-1$ 个脉冲扣除一个脉冲，即可实现分频系数为 $N-0.5$ 的分频器。脉冲扣除电路由异或门和一个 2 分频器（即 D

触发器）构成。电路组成如图 7-26 所示。

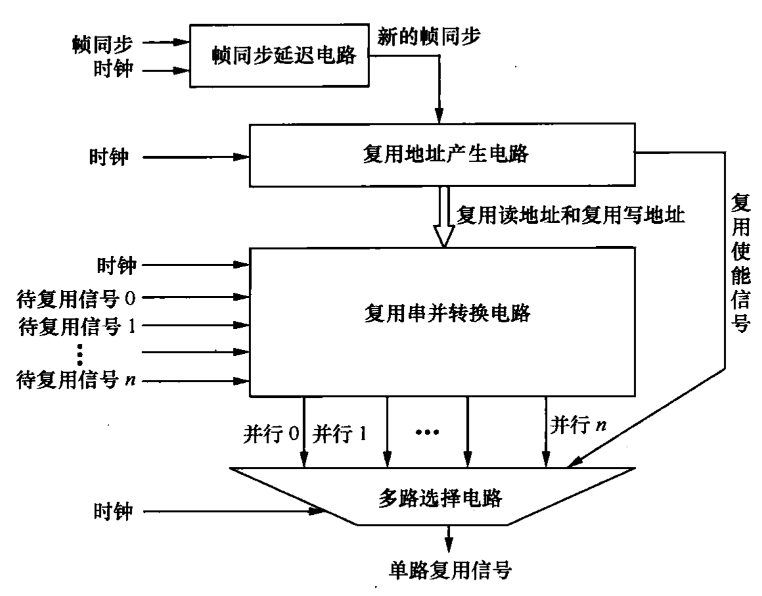


图 7-25　同步数字信号复用分解电路框图

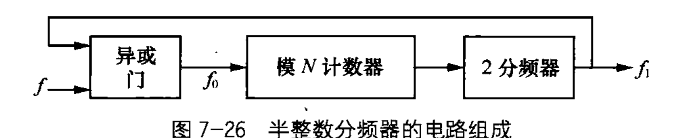


图 7-26　半整数分频器的电路组成

若要实现整数和半整数通用分频器，则可在半整数分频器原理的基础上，对异或门设置一个选通参数 sel，通过对异或门和计数器计数状态值的控制，实现同一个电路完成半整数及整数分频，如图 7-27 所示。当 sel 为 1 时，实现半整数分频，$f_1=f/(N-0.5)$；当 sel 为 0 时，实现整数分频，$f_2=f/N$。

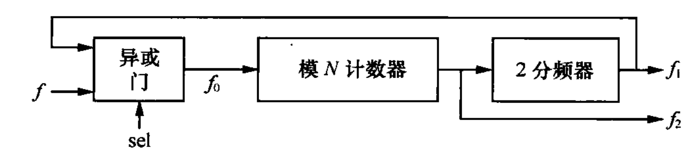


图 7-27　整数半整数分频器的电路组成

底层代码设计：

（1）异或门控制电路。

以下是异或门控制电路的 VHDL 程序设计。sel 设置为参数，当它为 1 时，c=a⊕b，实现半整数分频；当它为 0 时，c=b，实现整数分频。

```
library ieee;
use ieee.std_logic_1164.all;
entity control is
Port ( a,b: in std_logic;
    sel: in std_logic;
       c: out std_logic);
```

```
end control;
architecture Behavioral of control is
begin
          process(sel,a,b)
      begin
          if sel='1'then
              c<= a nor b;
          else
             c <= b;
          end if;
      end process;
end Behavioral;
```

（2）模 N 计数器参考第 4 章计数器部分，分频器从最高位输出。2 分频电路即 D 触发器，参考第 4 章触发器部分，这里不再叙述。

顶层设计可仿照第 4 章通过原理图输入的方法进行。

7.6.3 FPGA 滤波器电路

数字滤波器根据其冲击响应函数的时域特性，可分为有限长冲激响应（FIR）滤波器和无限长冲激响应（IIR）滤波器两种。FIR 滤波器被广泛应用于各类数字信号处理系统，它的系统总是稳定的，可以满足滤波器对幅度和相位特性的严格要求，避免模拟滤波器温漂和噪声等问题，易实现线性相位且易用硬件实现。IIR 滤波器系统比 FIR 滤波器系统易取得较好的通频带和阻带衰减特性，FIR 系统若要取得较好的衰减特性，一般要求系统函数 $H(z)$阶次要高，即滤波器长度 M 要大。本小节对 FIR 滤波器作简单介绍。

FIR 滤波器的系统函数为 $H(z)=\sum_{n=0}^{M-1}h(n)z^{-n}$，由系统函数可直接写出输入 $x(n)$和输出 $y(n)$之间关系的差分方程。设 $x(n)$是一个 M 点序列，则得 $y(n)=\sum_{i=0}^{M-1}x(i)h(n-i)$。就硬件实现而言，FIR 数字滤波器的基本结构是一个分节的延时线，每一节的输出加权累加，得到滤波器的输出。

通常在设计滤波器之前，应该先根据具体的应用确定一些技术指标。指标的形式一般在频域中给出幅度和相位响应。在确定了技术指标之后，就可以根据数学知识和滤波器的基本原理确定滤波器的模型来逼近给定的指标，可采用 Matlab 工具进行滤波器设计，得到滤波器抽头系数 $h(n)$。注意，MATLAB 中算出的系数 $h(n)$的值是一组浮点数，进行浮点值到定点值的转换（将 $h(n)$扩大 2^n 倍）。至此完成了一个滤波器设计的全过程。

下面利用 FPGA 来完成滤波器的硬件实现。FIR 滤波器主要是由乘加单元组成，利用 FPGA 硬件实现滤波的 TOP-DOWN 结构图，如图 7-28 所示。采用串行结构实现硬件即将输入数据直接与其对应的滤波系数相乘，将前一级乘积锁存，直接与后一级乘积累加，这样就可以节约硬件资源，提高执行速度。FIR 数字滤波器系统主要分为数据存储和数据运算两大模块。数据存储模块主要功能是以时钟去控制片选信号和地址译码，在 ROM 查找表中读出与 ROM 地址相对应的数据，即为滤波系数（定点数 $h(n)$），并将它与对应的输入信号同步输出至数据运算模块。数据运算模块主要功能就是完成输入信号与对应滤波系数的相乘和累加。

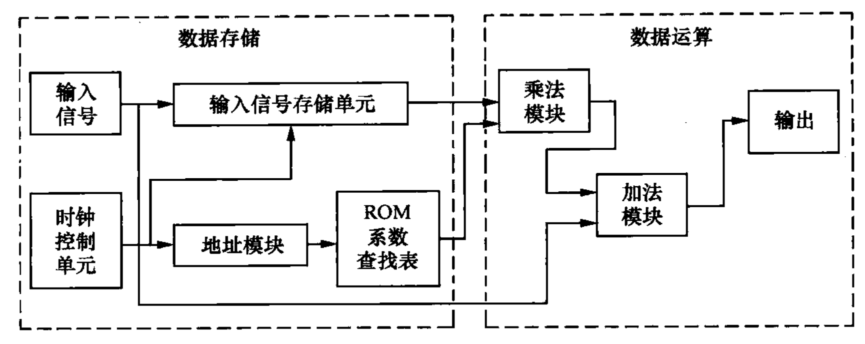


图 7-28 FIR 滤波器硬件实现 top-down 结构图

7.7 信号控制电路

7.7.1 可编程的交通信号灯控制电路

1. 功能要求

本系统要求一个由一条主干道和一条支道公路的汇合点形成的十字交叉路口的智能交通灯控制器，实现交通无人自动管理。要求是优先保证主干道的通畅，因此，平时处于“主干道绿灯，支道红灯”的状态，只有在支道有车辆要穿行主干道时，才将交通切换“主干道红灯，支道绿灯”的状态。此外，主干道和支道每次通行的时间不得低于 30s，而在两个状态交换过程出现的“主黄，支红”和“主红，支黄”状态，持续时间都为 4s。

2. 设计思路

本设计采用自顶向下（TOP-DOWN）的设计方法。该方法是一种从抽象到具体，从高层次到低层次逐步求精的分层次、分模块的设计方法，它是数字系统中最常用的一种设计方法，也是基于复杂可编程器件进行系统设计的主要方法。该方法首先从整体上规划了整个系统的功能和性能，然后对系统进行划分，以将其分解为规模较小、功能较为简单的局部模块，并确定他们之间的相互关系。这种划分过程可以不断地进行下去，直到划分所得到的单元可以映射到物理层为止。

通过交通灯控制系统的具体设计介绍了如何用该方法进行数字系统的设计，此设计方法同样适用于复杂数字系统的设计。根据交通信号灯控制的要求，把它分解为分频电路模块、计数秒数选择电路模块、倒计时控制电路模块和交通灯状态控制模块。分频电路模块以一个稳定的时钟让系统正常运转，它产生一些额外的输出信号，并将其作为其他几个模块的使能控制与同步信号。计数秒数选择模块以秒为单位倒计时，当计数值减为零时，主控电路改变输出状态，电路进入下一个状态的倒计时。其中，核心部分是交通灯状态控制模块。其原理框图如图 7-29 所示。

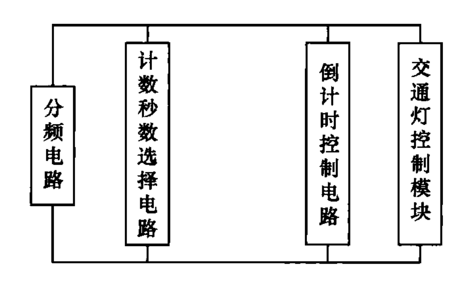


图 7-29 交通灯控制系统原理组成方框图

3．程序设计

程序设计系统由4个子电路构成：clk_gen（分频电路）、traffic_mux（计数秒数选择电路）、count_down（倒计时控制电路）、traffic_fsm（红绿灯信号控制电路）。当然计数秒数选择电路和倒计时控制电路可以合成一个定时器模块，产生4s和30s的两个定时。

下面以traffic_fsm 红绿灯信号控制电路的VHDL设计为例加以介绍，时钟脉冲发生模块（即分频模块）、计数秒数选择模块和倒计时控制模块由读者根据前面讲过的VHDL程序设计知识加以独立设计。有限状态机（Finite State Machine，简称FSM）是一类重要的时序电路，是许多数字系统的核心部件，也是实时系统设计中的一种数学模型，是一种易于建立的、以描述控制特性为主的建模方法，它可以应用于从系统分析到设计的所有阶段。有限状态机的优点在于简单易用，状态间的关系清晰直观。建立有限状态机主要有两种方法："状态转移图"和"状态转移表"，在此使用"状态转移图"来描述控制器的工作过程。根据主干道和支道的交通灯的变化情况，可以定义这样4种状态：S0为初始状态，主干道绿灯亮，支道红灯亮；S1为主干道黄灯亮，支道红灯亮；S2为主干道红灯亮，支道绿灯亮；S3为主干道红灯亮，支道黄灯亮。在状态转换的过程中，有这样一个规律：当进行状态转换时，定时器必须清零，此时clr为0，en为0；保持状态时，定时器一直进行定时，此时clr为1，en为1。

VHDL没有对状态机的描述规定一般的格式，但是为了使综合工具从VHDL描述识别并综合出状态机，需要遵循一定的编码风格，基本的FSM的编码风格是用CASE语句或其他等价方法来描述。一个有限状态机总是可以被分成次态译码、状态寄存器、输出译码3个模块。因此，有限状态机描述方式有如下3种：三进程描述、双进程描述和单进程描述，但是用VHDL描述状态机一般采用进程（process）描述：一个是时钟进程，控制状态机在时钟有效沿根据条件得到下一状态并进行状态迁移；另一个进程是组合进程，不受时钟控制，由输出相关的信号触发，该进程根据触发信号决定状态机的输出信号值。具体表述可参考第4章有限状态机一节。

根据交通信号灯控制器的状态转移图和有限状态机的描述方式，在此采用双进程描述方式，写出交通灯控制器的VHDL程序如下：

```
library ieee;
use ieee.std_logic_1164.all;
entity traffic_fsm is
port(clk,sb,reset: in std_logic;
   mr,my,mg,br,by,bg: out std_logic);
end jtd;
architecture behavioral of jtd is
   type state_type is (s0,s1,s2,s3);
   signal state : state_type;
begin
change_state:process (clk)                   --clk 为敏感信号量
   variable s: integer range 0 to 255;      --s 为秒定时器
   variable clr,en :bit;                    --clr,en 为秒定时器使能信号
   begin
   if reset = '1' then
      state<=s0;
```

```
    elsif (clk'event and clk='1') then
        if clr='0' then s:=0;                   --clr 为 0，定时器清零
        elsif en='0' then s:=s;                 --en 为 0，定时器保持不变
        else s:=s+1;                            --clr 为 1，en 为 1 时，正常计时
        end if;
        case state is
            when s0=>
                if s<30 then state<=s0;clr:='1';en:='1';
                                                --主干道绿灯持续时间未到 30s，维持 s0 状态
                elsif (s>=30 and sb='0') then state<=s0;clr:='1';en:='1';
                                                --已过 30s 但“支道无车”，维持 s0 状态
                elsif (s>=30 and sb='1') then state<=s1;clr:='0';en:='0';
                                                --已过 30s 且“支道有车”，转换到 s1 状态
                end if;
            when s1=>
                if s=4 then state<=s2;clr:='0';en:='0';   --已过 4s，转换到 s2 状态
                else state<=s1;clr:='1';en:='1';          --未到 4s，维持 s1 状态
                end if;
            when s2 =>
                if s<30 then state<=s2;clr:='1';en:='1';
                                                --支道绿灯持续时间未到 30s，维持 s2 状态
                elsif (s>=30 and sb='1') then state<=s2;clr:='1';en:='1';
                                                --已过 30s 但“支道有车”，维持 s2 状态
                elsif (s>=30 and sb='0') then state<=s3;clr:='0';en:='0';
                                                --已到 30s 且“支道无车”，转换到 s3 状态
                end if;
            when s3 =>
                if s=4 then state<=s0;clr:='0';en:='0';   --已过 4s，转换到 s0 状态
                else state<=s3;clr:='1';en:='1';          -- 未到 4s，维持 s3 状态
                end if;
        end case;
    end if;
end process change_state;
output_process:process (state)
    begin
    case state is
        when s0 => mr<='0';my<='0';mg<='1';br<='1';by<='0';bg<='0';
                                                --主干道绿灯亮,支道红灯亮
        when s1 => mr<='0';my<='1';mg<='0';br<='1';by<='0';bg<='0';
                                                --主干道黄灯亮,支道红灯亮
        when s2 => mr<='1';my<='0';mg<='0';br<='0';by<='0';bg<='1';
                                                --主干道红灯亮,支道绿灯亮
        when s3 => mr<='1';my<='0';mg<='0';br<='0';by<='1';bg<='0';
                                                --主干道红灯亮,支道黄灯亮
    end case;
end process output_process;
end;
```

程序说明：程序中 mg、my、mr 分别表示主干道上的绿灯、黄灯、红灯，bg、by、br 分别表示支道上的绿灯、黄灯、红灯，它们的值为“1”时表示灯亮，为“0”时表示灯灭；sb

表示支道传感器是否检测车辆存在，sb 为“1”时表示有车，为“0”时表示无车；定时器在控制器提供的计时信号 en 和清零信号 clr 的作用下完成定时功能，并向控制器提供 30s、4s 的计时信号；控制器是本系统的核心，它的作用是根据支道传感器和定时器的信号，判断、调整和控制整个系统的状态，并控制定时电路工作，提供适当的灯光控制信号。

7.7.2 十六路循环彩灯控制电路

此电路主要由 3 部分组成，其整体框图如图 7-30 所示。

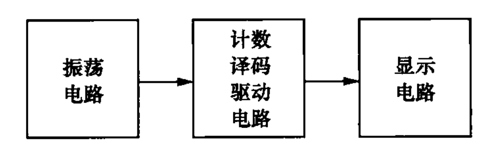


图 7-30 彩灯控制器整体框图

该循环控制电路由 555 定时器、同步 4 位二进制计数器 74LS163 和 4 线-16 线译码器 74LS154 组成。电路图如图 7-31 所示。

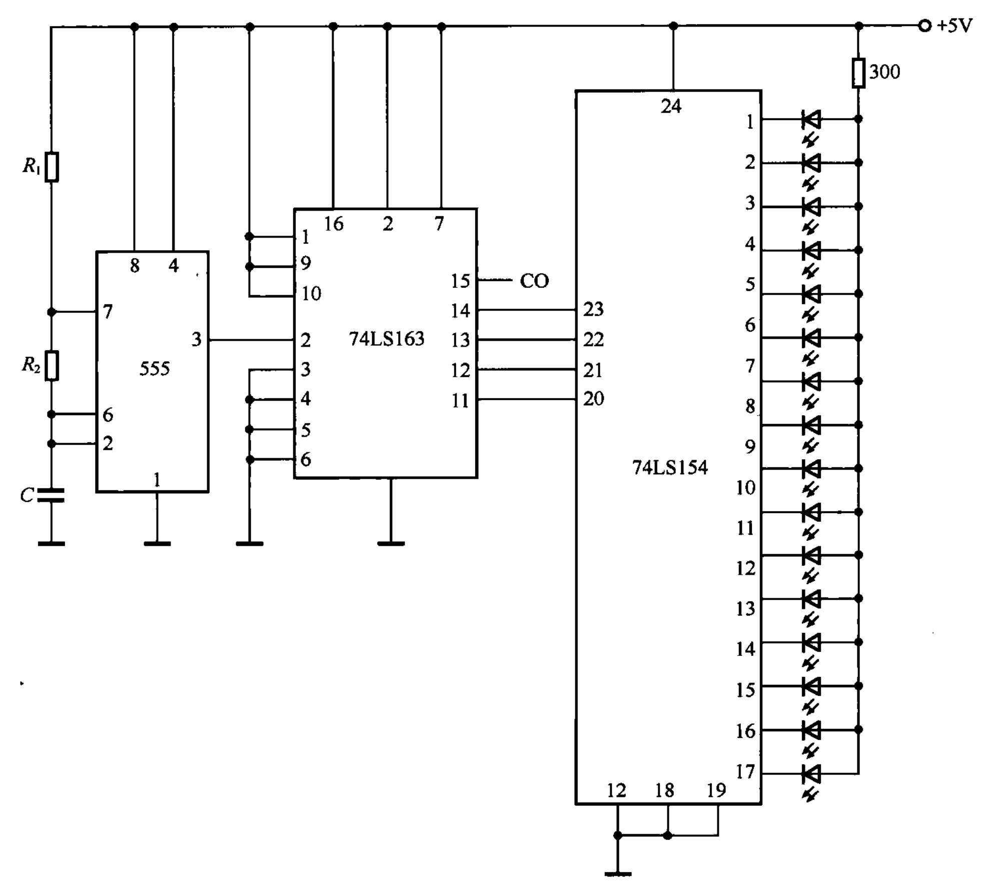


图 7-31 彩灯控制电路

电路中 555 定时器组成多谐振荡器，输出一定频率的脉冲。74LS163 位二进制计数器在输入周期性脉冲信号的作用下，输出的二进制数在 0000～1111 之间循环变化。通过 4 线—16

线译码器 74LS154，其 16 条输出线按照 74LS163 所输出的二进制数依次变化，从而与之相连的发光二极管依次点亮。彩灯循环显示频率快慢由 555 定时器产生的脉冲信号的频率决定，即调整电路中的电阻 R_1、R_2 及电容 C 的参数，可改变彩灯循环显示频率。

Multisim 软件仿真如图 7-32 所示，运行后可以看到发光二极管 LED1～LED16 循环点亮，通过调节电阻电容的值可以改变彩灯的循环频率。

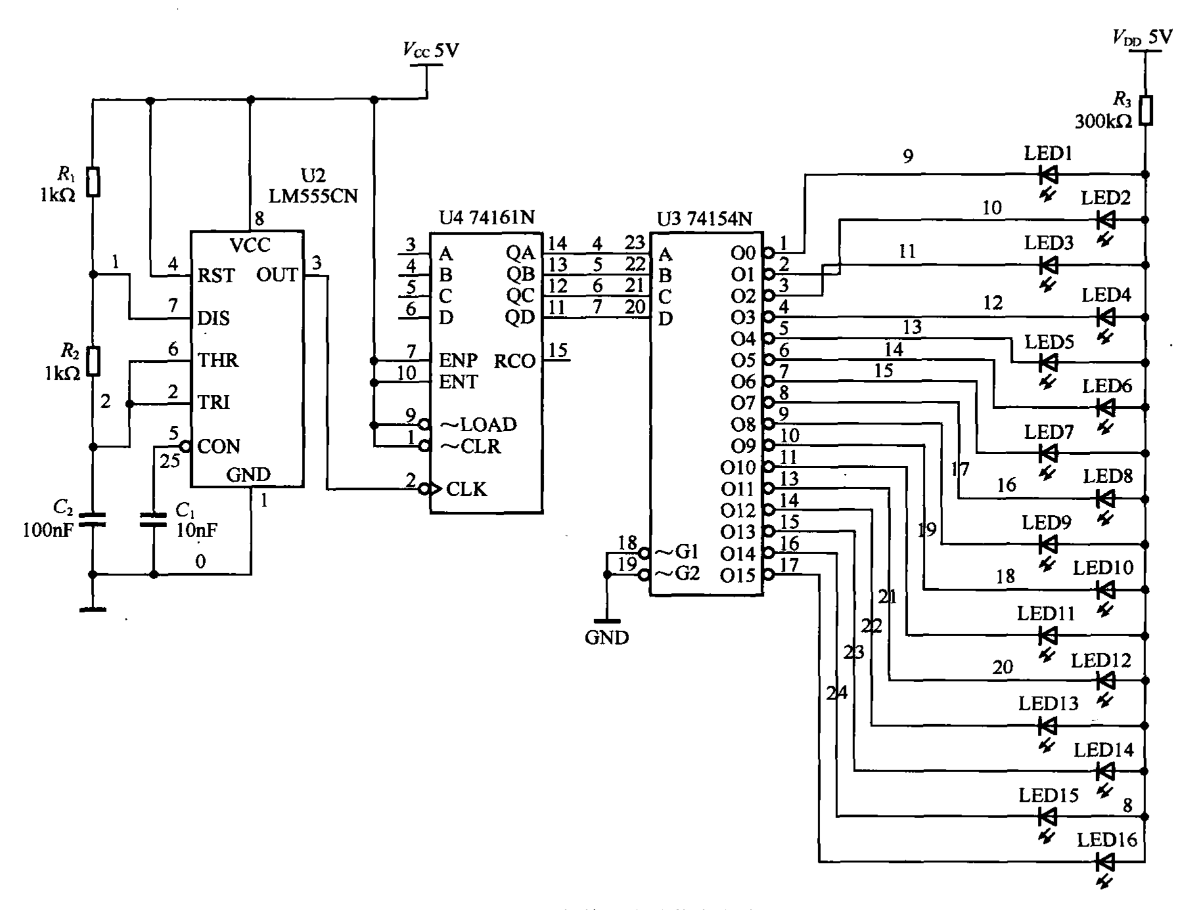


图 7-32 彩灯控制电路仿真电路

7.8 电源电路

现代电子设备中的电路使用了大量的半导体器件，这些半导体需要几伏到几十伏的直流供电，以便得到正常工作所必需的能源。大多数电子设备的直流供电方法是将交流电源经过变压、整流、滤波、稳压等变换为电子系统所需的稳定的直流电压。完成这种变换任务的电源成为直流稳压电源。

现代电子设备中使用的直流稳压电源有两大类：线性稳压电源和开关稳压电源。线性稳压电源亦称为串联整式稳压电源。它的稳压性能好、输出纹波小，缺点是需要使用体积和重量都比较大的工频变压器，而且稳定性较低。开关型稳压电源效率高、体积小、重量轻，缺点是输出的纹波及产生的电磁干扰比较大。开关电源和线性电源的成本随着输出功率的增加

而增长，但二者增长速率各异。通常，当输出功率较小时，线性电源的成本较低。而当线性电源成本在某一输出功率点上时，成本反而高于开关电源，这一点称为成本反转点。

本节将介绍直流可调稳压电源的设计和一种串联型开关稳压电源。

7.8.1 直流可调稳压电源的设计

直流稳压电源的作用是通过把 50Hz 的交流电变压、整流、滤波、和稳压，从而使电路变成恒定的直流电压供给负载。设计出的直流稳压电源应不以电网电压的波动和负载的变换而改变。

直流稳压电源的种类有很多，常用的是串联型直流稳压电源。由于集成技术的发展，集成稳压器件方便而可靠，逐渐代替了串联型直流稳压电源中的调整管及相关电路。

直流可调稳压电源的完整电路如图 7-33 所示。

直流电源通常从市电取电，首先通过变压电路把 220V、50Hz 的单相交流电先降压，变成所需的交流电，然后再整流。整流采用的二极管桥式整流电路。根据经验，一般滤波电路常用的滤波电容有 2.2mF 和 1.1mF 两种，但要注意它的耐压值要大于电路中所承受的电压，这里采用的是 2.2mF 滤波电容。集成稳压电源的核心器件是 LM317，在实际应用中要注意加装散热片。

为了保护集成器件在接反的状态下不被烧毁，在输入、输出端之间以及输出与调节端之间分别接反以保护二极管 1N4003。另外，电容 C_2 和 C_3 分别为去抖和滤波作用。C_2 并联在可变电阻器两端，可防止可变电阻器在调节过程中由于抖动而产生的谐波，一般经验值为 10μF。C_3 为输出侧二次滤波，其目的是去掉输出电压波形中细小的纹波。C_1 与 C_3 的关系一般为 20 倍左右。通过电路仿真可以看到示波器输出为稳定的直流电压，通过调节 R_1，输出的电压范围为 1.3V～16V。

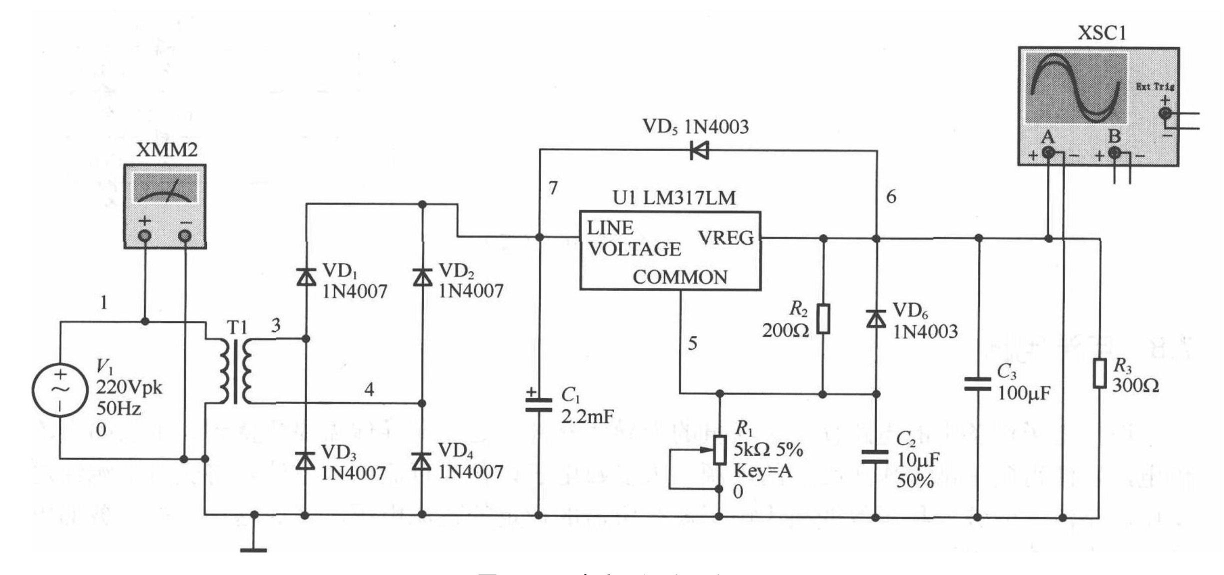


图 7-33 直流可调稳压电源

7.8.2 串联型开关稳压电源

随着开关稳压集成电路的发展，开关稳压电路得到了非常广泛的应用。目前中功率以上

的稳压电源都是开关型稳压，而且迅速地延伸到小功率稳压的领域。开关稳压有许多线性稳压所不具备的优点：一是效率高，线性稳压电路的效率只能达到30%～60%，而开关稳压电路的效率一般都能达到80%以上，近些年来推出的开关稳压集成电路的效率多在90%以上；二是不仅仅可以设计成降压型（即输出电压低于输入电压），还可以设计成升压型和输出电压极性倒换型等多种形式，甚至在一个稳压电源中都可以有多种电压、多种形式的输出，从而构成一个完整的电源系统；三是有的开关稳压电源甚至不需要变压器，直接将220V交流整流后变换成稳定的低压电流，使得电源的体积大大缩小。

开关稳压是将不稳定的直流，通过变换振荡器变成较高频率的矩形波，再经过高频整流和电感、电容储能滤波，变成另一稳定的直流。由控制电路对振荡器输出电压的脉冲宽度或频率进行控制，从而实现稳定直流输出电压的目的。

开关型稳压电路分为脉冲宽度调制式、脉冲频率调制式和谐振式3类，目前集成开关稳压器大多采用脉冲宽度调制（PWM）工作方式。下面介绍一种串联型开关稳压电源。

本开关电源是用555定时器组成的脉宽调整电路构成串联式稳压源，如图7-34所示。

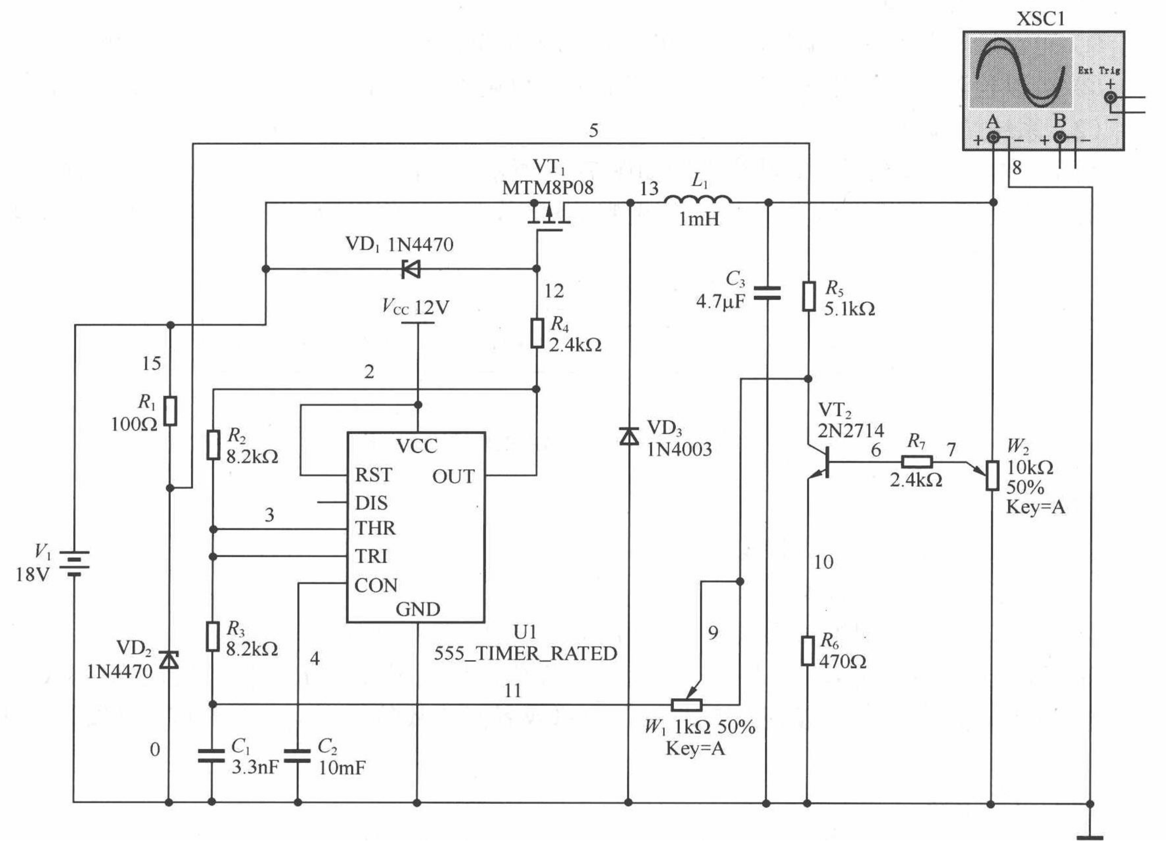


图7-34 串联型开关稳压电源电路

电路的控制线性和对称性较好。在输入直流电压为18V±10%时，输出直流为3～12，连续可调；电流为0～5A；电压稳定度≤15%；纹波电压 $V_{p\text{-}p}$≤25mV；开关工作频率7kHz～17kHz；效率约为67%。

本电路与常见的555脉宽调制器相比，是在 C_1 与2、6脚之间接了 R_3，减小 C_1 上的电

压振幅；BG_2、R_5、R_6、R_7、W_1、W_2组成输出电压取样、放大电路，调节W_1和W_2，可改变反馈电压的大小，从而改变开关方波的占空比，改变输出直流电压的大小；BG_1采用P沟道TMOS管MTM8P08；储能电感约在1mH；VD_3为续流管。

7.9 设计范例

通过前面电子设计中各部分电路的介绍，已经对电子设计的基本流程有了一定的了解。但在实际的电路设计中，由于实际环境的复杂性，将会遇到各种各样的问题，这里仅仅给出几个例子，抛砖引玉，供读者学习，为掌握现代电子电路的设计方法奠定基础。

7.9.1 数字式电缆对线器

1．技术指标

（1）整体功能要求。

① 可在远端预设芯线编号，近端测量出对应的芯线并且以数字显示出电缆芯线编号。

② 可以检测到电缆芯线的短路或开路故障。可由人工单线接入测试，亦可自动测试。

（2）系统结构要求。

数字式电缆对线器的系统结构框图如图7-35所示。其中远端编号器用于给被测电缆处于远端的芯线编号，电缆对线器在电缆近端测出各芯线与远端的对应关系并以数字显示出近端各芯线的编号数码。

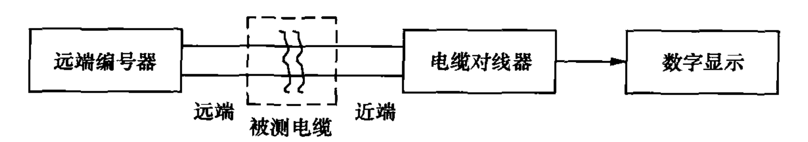


图7-35 数字式电缆对线器的系统结构框图

（3）电气指标。

① 对线器一次可接入的芯线数量为8根。

② 芯线编号显示方式为2位数码。

③ 显示及刷新时间为2s刷新1次，显示数码时间不少于1s。

④ 测试方式为远端编号，接好芯线后不再操作，近端用人工方式逐一选择被测芯线。

⑤ 电缆故障报警：当发现某条芯线有短路或者开路故障时，发出报警信号（发光二极管亮）。

（4）设计条件。

① 电源条件：直流稳压电源提供±5V电压，稳压电源设计可参考本章7.8节。

② 被测电缆长度为1000m，芯线直径为0.4mm，直流电阻为148Ω/km，绝缘电阻为2000MΩ/km。测试前有一根芯线远、近端均已明确，用其作为测试地线。

2．整体方案设计

电缆对线器的原理框图如图7-36所示。

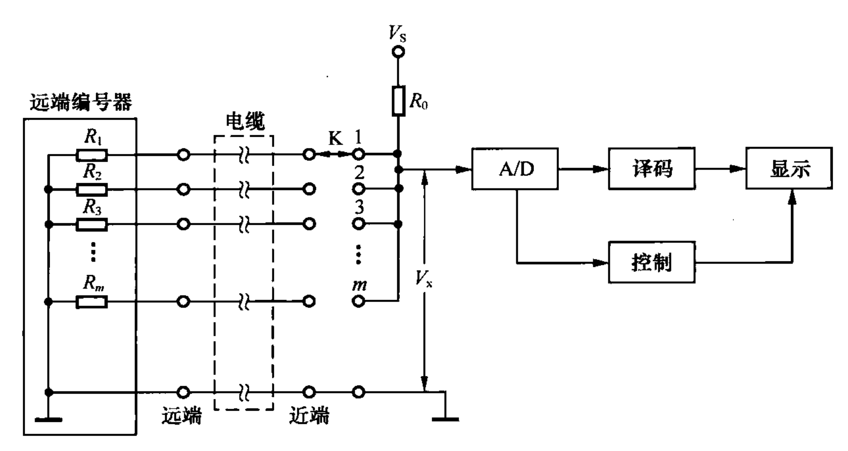


图 7-36 电缆对线器的原理框图

先在远端把被测电缆芯线分别与远端编号器中的 R_1～R_m 连接，并约定与 R_1 连接的为 1 号线。依次类推，远端为 m 根芯线编号。若定义远端编号器中某一电阻为 R_x，在忽略电缆导线电阻的情况下，在近端可得

$$V_x = \frac{R_x}{R_0 + R_x} V_s \tag{7-8}$$

由于 R_1～R_m 均是事先选定的，所以，当近端的开关 K 位于不同位置时，都可事先算出 m 个 V_x 值。这些 V_x 值经 A/D 转换成为一组数量化的数字值。可事先将这 m 个数字值建立一张译码表，表中 m 个量化后的数字值对应着 m 个导线号码。例如，当近端开关 K 位于 2 的位置时，得到一个 V_x 值及经 A/D 转换后的量化值，将这一量化值译码为数字，该数字就显示成为芯线编号。

（1）A/D 转换器输出位数及转换阶梯。

为提高转换精度，取 ADC 的高 5 位输出，则$\triangle V_s$=5/32=0.156V。则输入模拟量和输出二进制量化值之间的关系如表 7-1 所示。

表 7-1　　A/D 输入模拟量和输出二进制量化值之间的关系

输入模拟量		A/D 输出量化值				
$x\triangle V_s \leqslant V_x < (x+1)\triangle V_s$		2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}
0	$0 \leqslant V_0 < 0.156$	0	0	0	0	0
1	$0.516 \leqslant V_1 < 0.313$	0	0	0	0	1
2	$0.313 \leqslant V_2 < 0.469$	0	0	0	1	0
3	$0.469 \leqslant V_3 < 0.625$	0	0	0	1	1
4	$0.625 \leqslant V_4 < 0.781$	0	0	1	0	0
5	$0.781 \leqslant V_5 < 0.938$	0	0	1	0	1
6	$0.938 \leqslant V_6 < 1.094$	0	0	1	1	0
7	$1.094 \leqslant V_7 < 1.25$	0	0	1	1	1
8	$1.25 \leqslant V_8 < 1.406$	0	1	0	0	0
9	$1.406 \leqslant V_9 < 1.563$	0	1	0	0	1

续表

输入模拟量		A/D 输出量化值				
$x\triangle V_s \leqslant V_x<(x+1)\triangle V_s$		2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}
10	$1.563\leqslant V_{10}<1.719$	0	1	0	1	1
11	$1.719\leqslant V_{11}<1.875$	0	1	0	1	1
12	$1.875\leqslant V_{12}<2.031$	0	1	1	0	0
13	$2.031\leqslant V_{13}<2.188$	0	1	1	0	1
14	$2.188\leqslant V_{14}<2.344$	0	1	1	1	0
15	$2.344\leqslant V_{15}<2.5$	0	1	1	1	1
16	$2.5\leqslant V_{16}<2.656$	1	0	0	0	0
17	$2.656\leqslant V_{17}<2.813$	1	0	0	0	1
18	$2.813\leqslant V_{18}<2.969$	1	0	0	1	0
19	$2.969\leqslant V_{19}<3.125$	1	0	0	1	1
20	$3.125\leqslant V_{20}<3.281$	1	0	1	0	0
21	$3.281\leqslant V_{21}<3.438$	1	0	1	0	1
22	$3.438\leqslant V_{22}<3.594$	1	0	1	1	0
23	$3.594\leqslant V_{23}<3.75$	1	0	1	1	1
24	$3.75\leqslant V_{24}<3.906$	1	1	0	0	0
25	$3.906\leqslant V_{25}<4.063$	1	1	0	0	1
26	$4.063\leqslant V_{26}<4.219$	1	1	0	1	0
27	$4.219\leqslant V_{27}<4.375$	1	1	0	1	1
28	$4.375\leqslant V_{28}<4.531$	1	1	1	0	0
29	$4.531\leqslant V_{29}<4.688$	1	1	1	0	1
30	$4.688\leqslant V_{30}<4.844$	1	1	1	1	0
31	$4.844\leqslant V_{31}<5$	1	1	1	1	1

（2）R_0 的取值。

电缆线径为 0.4mm 时，芯线电阻 r 为 148Ω/km，芯线绝缘电阻 R_d 为 2000MΩ/km，如图 7-37 所示。

$R_0 \geqslant 10\times 2^n r(n=5, r=148\Omega)$ 即 $R_0 = 10\times 2^5 \times 148 =$ 47.36kΩ 取标称值 51k 的阻值。

（3）远程编码器电阻选定。

根据公式 $R_x = \dfrac{2x+1}{2(2^n - x)-1} R_0 = K_x R_0$ 计算电阻值 $R_1 \sim R_m$，其中 R_0 和 R_{31} 留下作故障判断用，分别为短路和断路，如表 7-2 所示。

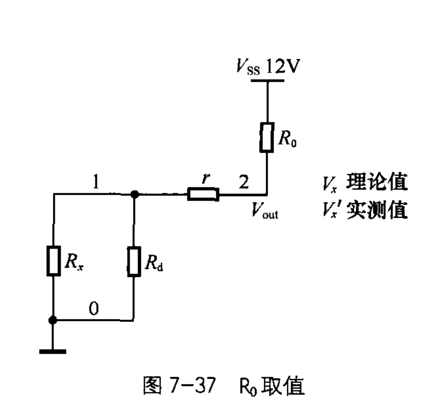


图 7-37　R_0 取值

表 7-2 R_1～R_m的阻值

x	K_x	R_x	标称取值	V_x	x	K_x	R_x	标称取值	V_x
0				0	16	1.06	54.29	51k+3.3k	2.58
1	0.05	2.51	2.4k	0.22	17	1.21	61.55	62k	2.74
2	0.08	4.32	4.3k	0.39	18	1.37	69.89	68k+1.8k	2.89
3	0.12	6.26	6.2k	0.54	19	1.56	79.56	75k+4.7k	3.05
4	0.16	8.35	8.2k	0.69	20	1.78	90.91	91k	3.20
5	0.21	10.58	10k	0.82	21	2.05	104.43	100k+4.3k	3.36
6	0.25	13.00	13k	1.02	22	2.37	120.79	100k+22k	3.53
7	0.31	15.61	15k	1.14	23	2.76	141.00	110k+30k	3.66
8	0.36	18.45	18k	1.30	24	3.27	166.60	150k+16k	3.82
9	0.42	21.53	22k	1.51	25	3.92	200.08	200k	3.98
10	0.49	24.91	22k+3k	1.64	26	4.82	245.73	200k+47k	4.14
11	0.56	28.61	27k+1k	1.77	27	6.11	311.67	270k+43k	4.30
12	0.64	32.69	33k	1.96	28	8.14	415.29	360k+56k	4.45
13	0.73	37.22	36k+1k	2.10	29	11.80	601.80	300k×2	4.61
14	0.83	42.26	43k	2.29	30	20.33	1037.00	560k+470k	4.76
15	0.94	47.91	39k+8.2k	2.40	31				5

3．单元电路设计

（1）A/D 转换电路：参考本章 7.4.2 节 A/D 转换电路的设计。

（2）码制转换，将 ADC 输出的二进制码转换为 BCD 码用于显示，用 EEPROM 28C64 实现，将 0～31 码表存储于该 PROM 中，如表 7-3 所示。

表 7-3 ROM 中存储的 0～31 码表

	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
00000000	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
00000016	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

（3）显示电路如图 7-38 所示，采用 7448 引脚。

（4）断路和短路报警。如图 7-39 所示，断路时，ADC 输出 D_4、D_3、D_2、D_1、D_0 全为 1，短路时全为 0，用门电路实现报警，驱动发光二极管。

$$F_{断路} = D_7 D_6 D_5 D_4 D_3 = \overline{\overline{D_7 D_6 D_5 D_4 D_3}} \quad (7\text{-}9)$$

$$F_{短路} = \overline{D_7 D_6 D_5 D_4 D_3} = \overline{D_7 + D_6 + D_5 \cdot D_4 + D_3} = \overline{\overline{\overline{D_7 + D_6 + D_5 \cdot D_4 + D_3}}} \quad (7\text{-}10)$$

4．测试与调整

（1）显示电路调测。

将万用表拨至电阻×1 档，红表笔接地，黑表笔接至 7448 引脚，则数码管对应灯亮。

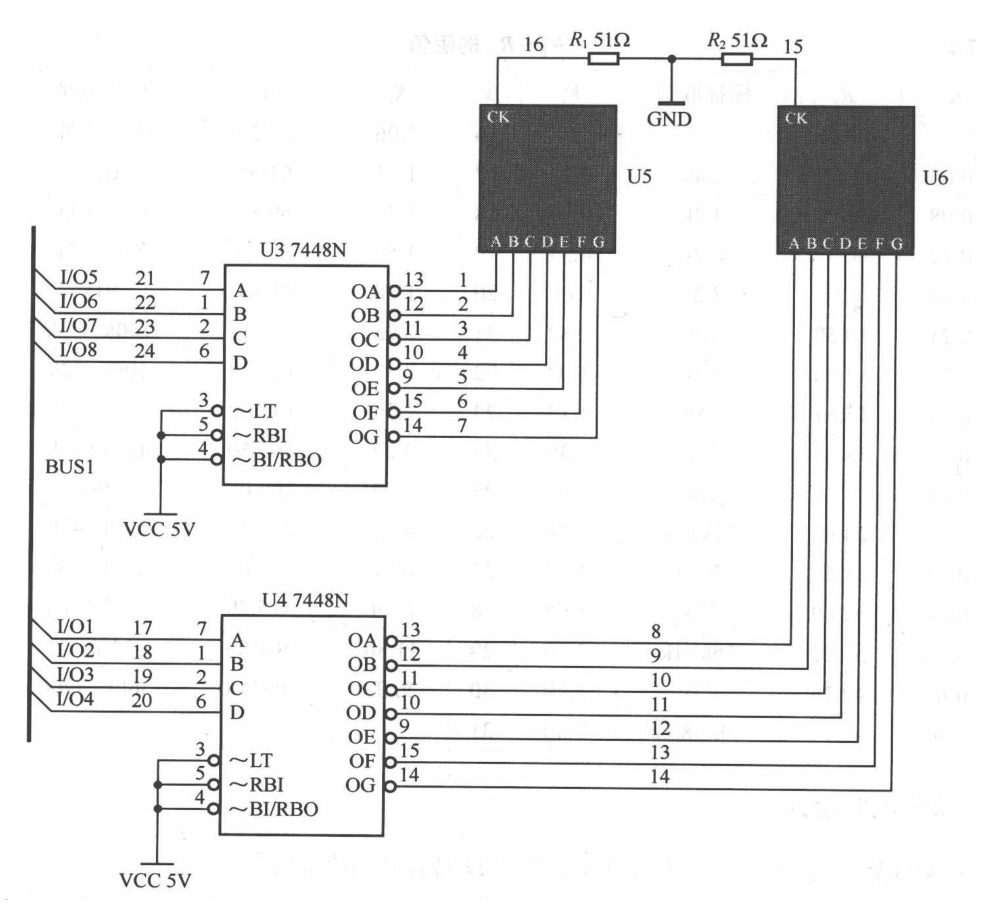


图 7-38 显示电路

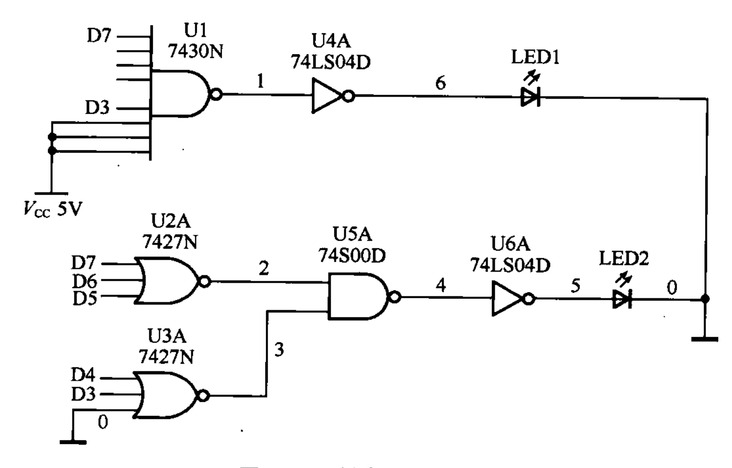


图 7-39 断路和短路报警

（2）A/D 转换电路调测。

依次拨动 8 位拨码开关的 1～8，将示波器接入电路。

依次记录下各测量数据，如表 7-4 所示，小写部分为测量值。

表 7-4 数码管显示数值

编号	远端编码器阻值	V_x	A/D 输出	显示
1	$8.2K_{(8.8K)}$	0.80V	00100	04
2	$200K_{(198K)}$	3.68V	11001	25
3	$10K_{(10.2K)}$	0.95V	00101	05
4	$4.3K_{(5K)}$	0.55V	00010	02
5	$15K_{(15.6K)}$	1.21V	00111	07
6	$22K_{(21.8K)}$	1.59V	01001	09
7	$0_{(短路)}$	0	00000	00
8	∞	5.12V	11111	31

经测试，手动测量达到设计指标，自动测量也达到设计指标。

7.9.2 温度测量仪

1．技术指标

（1）系统整体功能要求。

温度测量仪能够测量和显示测量的温度值，当温度超过设定的值后，发出超温的指示或报警。报警温度的设定可根据需要自定。

（2）系统结构要求。

温度测量仪的整体框图如图 7-40 所示，其中 S1 为系统复位按键，S2 为报警温度设定。

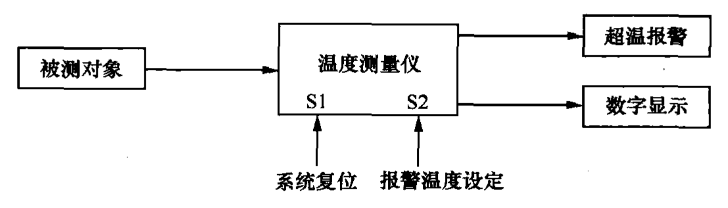


图 7-40 温度测量仪的整体方案图

（3）电压指标。

① 温度测量范围：0℃～99℃

② 显示精度：1℃

③ 测温灵敏度：20mV/℃

④ 显示采用四位数码管

⑤ 温度报警采用 LED 发光二极管或蜂鸣器

⑥ 报警温度可以任意设定

2．整体方案设计

温度测量仪是通过温度传感器对被测对象的稳定变化情况进行测量和监视，传感器输出的不同电压，经放大相应的倍数形成不同的模拟电压，经 A/D 转换，译码后送入数码管，温度数值即可显示出来。

其原理及整体方框图如图 7-41 温度测量仪整体方框图所示。整个系统电路在面包板上实现，其中核心模块为 A/D 转换部分及数字显示部分。

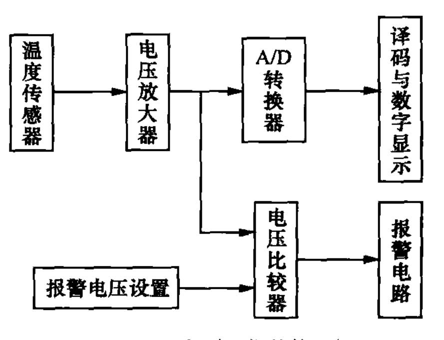


图 7-41　温度测量仪整体方框图

3. 单元电路设计

（1）预报警电路。

① 整体功能要求。基本电路中的电压报警电路中的标准参考电压 V_g 实现步进式可控过程，即当 $V_t > V_g$ 时，发光二极管显示报警。

② 整体设计思路。用按键脉冲作为二进制加/减计数器的时钟信号，计数器输出值作为 D/A 转换输出对应数值的温度电压，这个温度加在比较器的同相端与测量的温度电压作比较，大于测量电压即报警。

③ 预报警电路整体设计框图如图 7-42 所示。

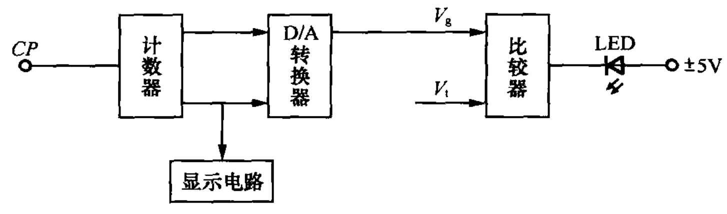


图 7-42　预报警电路整体设计框图

注意：图中的计数器应为 8 位计数器，因系统要求的温度测试范围为 0℃～99℃时，可用两片 74191 异步级联。计数器的输出作为 D/A 转换电路的数据输入而后转换为相应的模拟电压输出。显示电路显示的是计数器输出的值，为了简化该电路，又能正确反映计数值，可采用多个发光二极管分别显示各位的输出。

④ 预报警电路单元电路的设计。

● 时钟信号即按键脉冲稳定电路的设计。

为了稳定并正确地反映按键脉冲的计数值，采用了具有稳态的基本 SR 触发器的设计，该电路具有锁存功能，能将单脉冲锁存一段时间，其基本电路如图 7-43 所示。

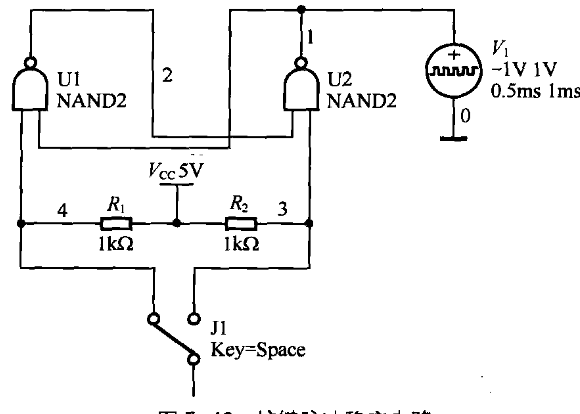


图 7-43　按键脉冲稳定电路

- 二进制加/减计数器的设计。

考虑到计数器的可控灵活性，选用 4 位加减计数器 74191，用另一按键控制其加减，将两片 74191 异步级联便可得到一个技术范围为 0～255 的二进制计数器，鉴于实际需要的计数范围为 0～99，可以将其高位的最高位悬空不用，其具体的级联电路图如图 7-44 所示。

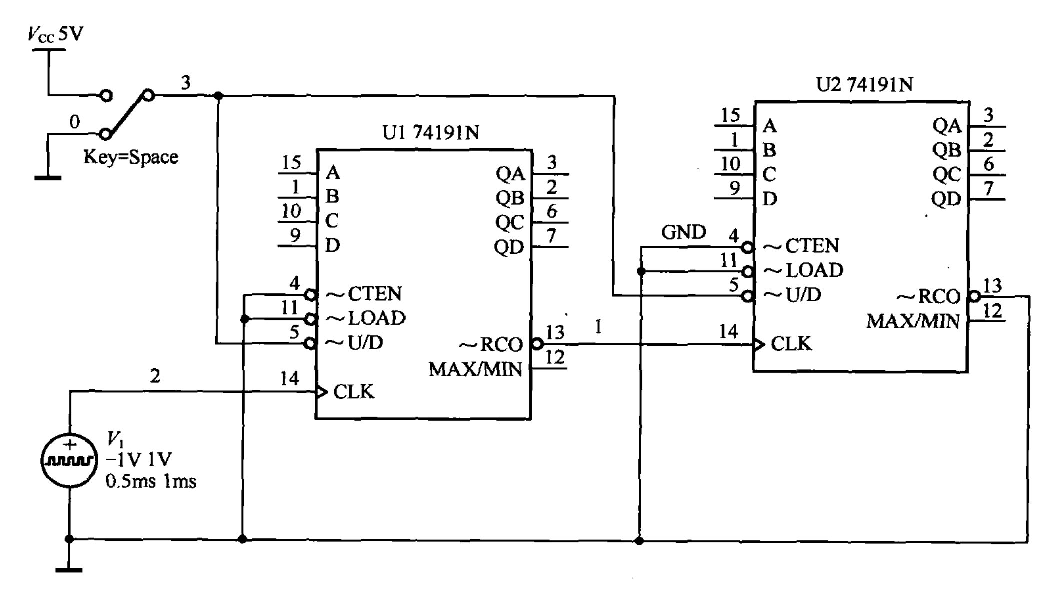


图 7-44 二进制加/减计数器

- 计数值显示电路的设计。

为了简化这个电路，显示电路选用了 7 个发光二极管分别显示 7 位计数器输出值，等价于比较温度 V_g 的摄氏值。为了保护发光二极管，在其负极各串联一个 1kΩ的电阻，其电路如图 7-45 所示。

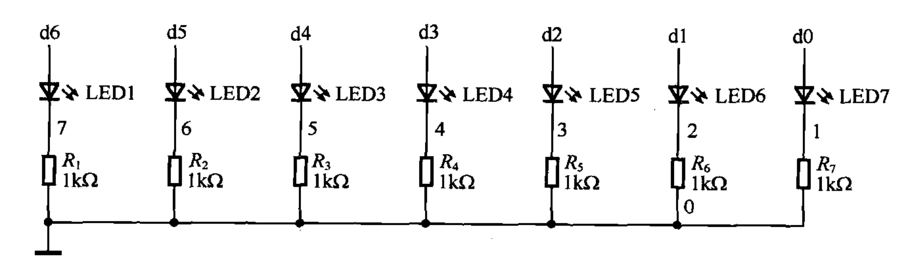


图 7-45 计数值显示电路

- D/A 转换电路的设计。

D/A 转换电路设计可以参考本章 7.4.1 节的内容。

- 电压比较报警电路。

事实上，该比较报警电路与温度报警电路是一样的，原理相同，只不过此时的要求略有不同，是当 $V_o>V_t$ 时 LED 发光报警，因此发光二极管 LED 的机型接法应做些改变，如图 7-46 所示。

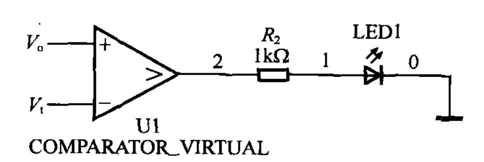


图 7-46 电压比较报警电路

- 预报警电路整体电路图。

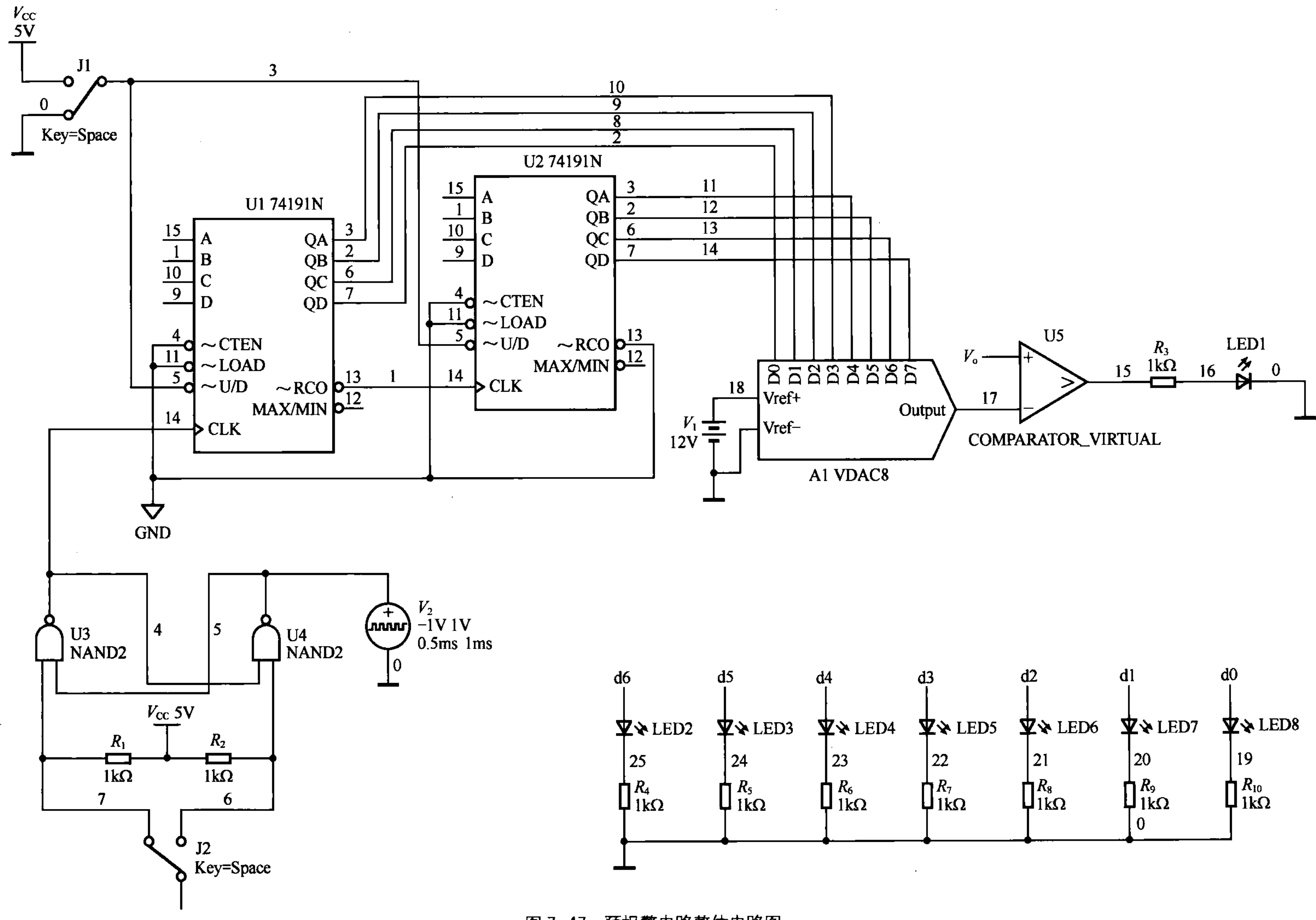


图 7-47 预报警电路整体电路图

（2）温度传感器及其放大电路的设计。

LM35 主要特性：灵敏度为 10mV/℃，常温下测温精度为±0.5℃以内，消耗电流最大也只有 70mA，自身发热时测量精度影响在±0.1℃以内。

设计思路：

因为用户要求测温灵敏度 20mV/℃，而 LM35 的灵敏度为 10mV/℃的电压输出型温度传感器，因此传感器温度变换后应有一个同相 2 倍的电压放大电路，这部分电路可简单地运用运算放大器 LM324 来实现。

温度变换及其电压放大模块电路图如图 7-48 所示。

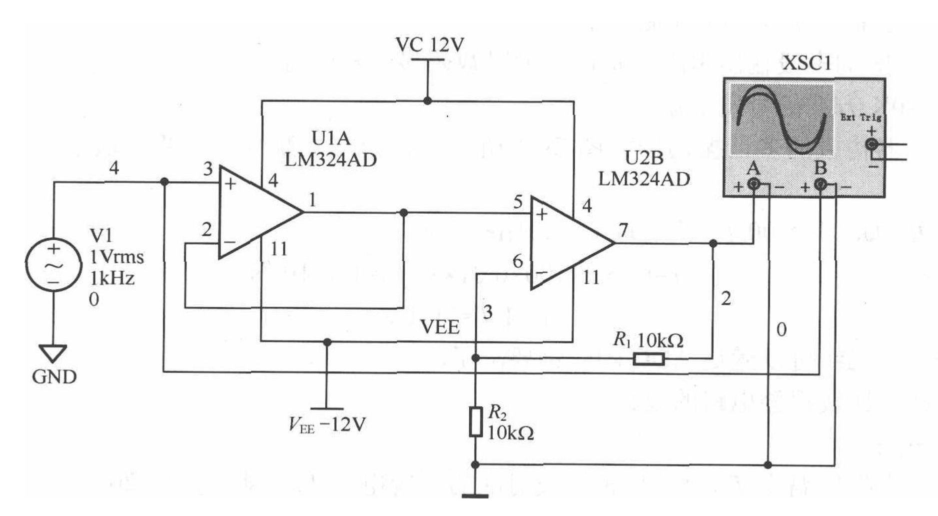


图 7-48　电压放大模块电路图

图中的跟随电路 U1A 是为了避免后续电路对 V_t 的过多影响而增设的电压跟随器，以保证 V_t 能真实地反映温度场的正确温度。

电路的放大倍数可以通过 $1+R_1/R_2$ 求得，如图 7-49 所示，当 $R_1=R_2$ 时，可以看到电压放大倍数为 2 倍，在测量温度时将输入信号接 LM35 温度传感器即可。

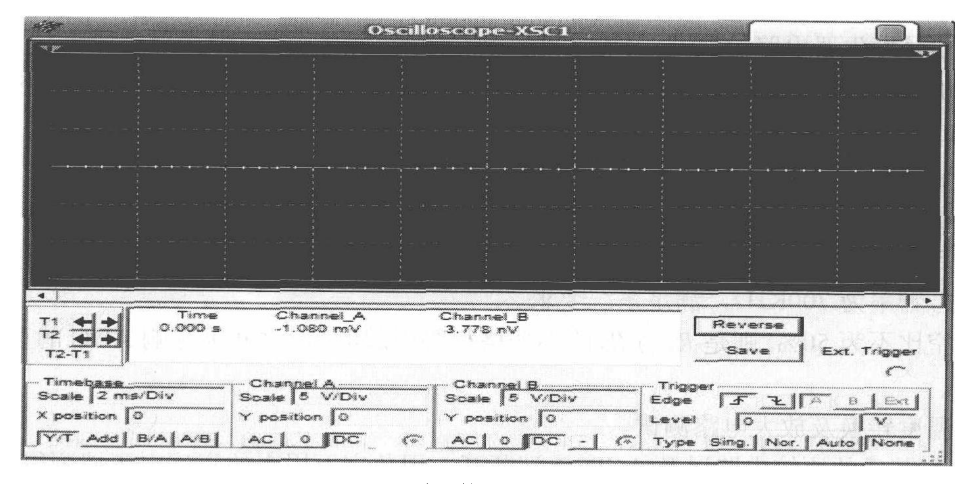


图 7-49　输入输出电压仿真波形图

（3）A/D 转换电路及数字显示电路的设计。

设计思路：

将 V_t 的模拟电压送入 A/D 转换器的输入端，转换为二进制码，用该码作为存储器 EEPROM 的地址信号，将事先预置在存储单元的温度值取出，经译码显示电路将数字显示出来。

模数转换及数字显示电路如图 7-50 所示。

图 7-50 中 ADC 的 SOC 端需要一个 640K 左右的方波信号，若不提供信号发生器，可以采用 555 定时器制成自激多谐振荡器来实现。而输入端 V_t 可选择 V_{in}，因此地址为 000。

27c64-12p 内应预先存储有地址 00H～63H 所对应的摄氏温度值，即 00～99，而 ADC 的输出则作为寻址信号即可得到温度值。

两片 7448 的七段输出引脚分别送至七段数码显示管即可。

（4）640K 方波信号发生器。

用定时器构成的多谐振荡器如图 7-51 所示，它无需加激励，只要接通电源就可以输出方波。

其中 $R_a=R_b$，取 100Ω，则若取 C=0.01μF，则

$$T=0.7\times2\times100\times0.01\times10^{-6}\text{s}=1.4\times10^{-6}\text{s} \tag{7-11}$$

$$F=1/T=714\text{Hz} \tag{7-12}$$

这个频率已经可以满足 ADC 的时钟要求了。

（5）超限比较报警电路的设计。

设计思路：

设定一个报警温度 T_g，将 T_g 折算成对应的比较电压 V_g，即 $V_g=T_g\times20\text{mV/℃}$。要使得当 $V_t>V_g$ 时，电路报警，可将两电压通过一个电压比较器后经发光二极管显示是否报警。

报警电路图如图 7-52 所示。

（6）整体电路图。

综合（1）～（5）可以得出该温度测量系统的整体电路图如图 7-53 所示。实际测量中 V_g 连接预报警电路的输出。

4．测试与调整

（1）信号发生器电路的测试。

按整机电路图连接好电路之后，将稳压电源引至面包板的电源端子上，断开 555 定时器 u_0 端与 ADC CLOCK 端的连接。

先将输出连到示波器观察 u_0 的输出波形，如果发现没有预先的方波输出。首先测电源与地线的电压是否正确，其次看 R_a、R_b 的分压是否有效，发现均无问题。最后再检查电路的接线，比如 555 定时器 2 与 6 端未相连，就会出现上述问题。更正错误后，输出波形就为接近方波，信号频率为 700KHz，满足系统要求。

而占空比不为 50%，则是 R_a 与 R_b 的不严格相等引起的，但这并不影响 ADC 的正常工作，测试通过。

（2）温度转换及放大电路测试。

将 V_t 与其后的比较电路以及 ACINO 端断开，用数字万用表的直流电压档测得 LM35 的中间脚为 0.29V，因 LM34 灵敏度为 10mV/℃，0.29V 即相当于室温 29℃，这与当时的实验



图 7-50 模数转换及数字显示电路

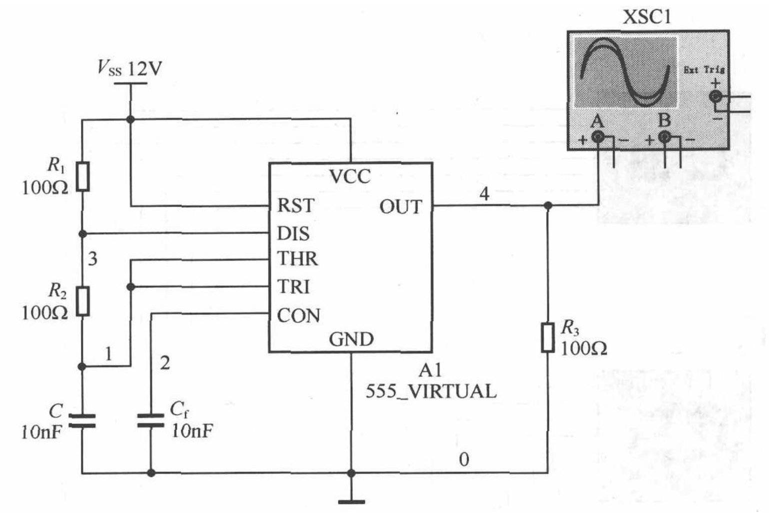


图 7-51　多谐振荡器电路

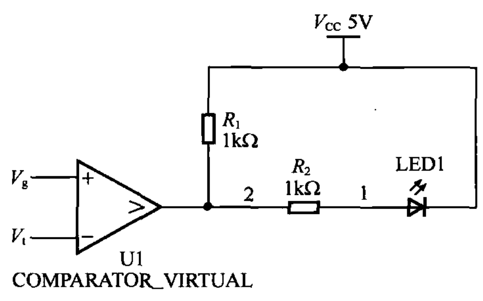


图 7-52　报警电路图

室温一致，表明 LM35 正常工作。

继续将运算放大器 LM324 的放大端输出电压值测出得 U_o=0.60V，严格放大两倍则 U_o 应为 0.58V，因此该放大器存在较大误差，可将图 7-53 中的 Ra 串联上一个总阻值为 10kΩ的电位器，只需调整至 U_o=0.58V 时，N 即严格等于 2 了。

（3）A/D 转换电路及数字显示电路的调测。

7.9.3　宽带直流放大器设计

1．技术指标

技术指标有以下几种：

（1）最大电压增益 AV≥60dB，输入电压有效值 V_i≤10 mV。AV 可在 0～40dB 范围内手动连续调节。在 AV＝60dB 时，输出端噪声电压的峰－峰值 V_{ONPP}≤0.3V。

（2）最大输出电压正弦波有效值 V_o≥2V，输出信号波形无明显失真。

（3）3dB 通频带 0～5MHz；在 0～4MHz 通频带内增益起伏≤1dB。

（4）放大器的输入电阻≥50Ω，负载电阻(50±2)Ω。

（5）最大输出电压正弦波有效值 V_o≥10V，输出信号波形无明显失真。

（6）电压增益 AV 可预置并显示，预置范围为 0～60dB，步距为 5dB（也可以连续调节），放大器的带宽可预置并显示（至少 5MHz、10MHz 两点）。

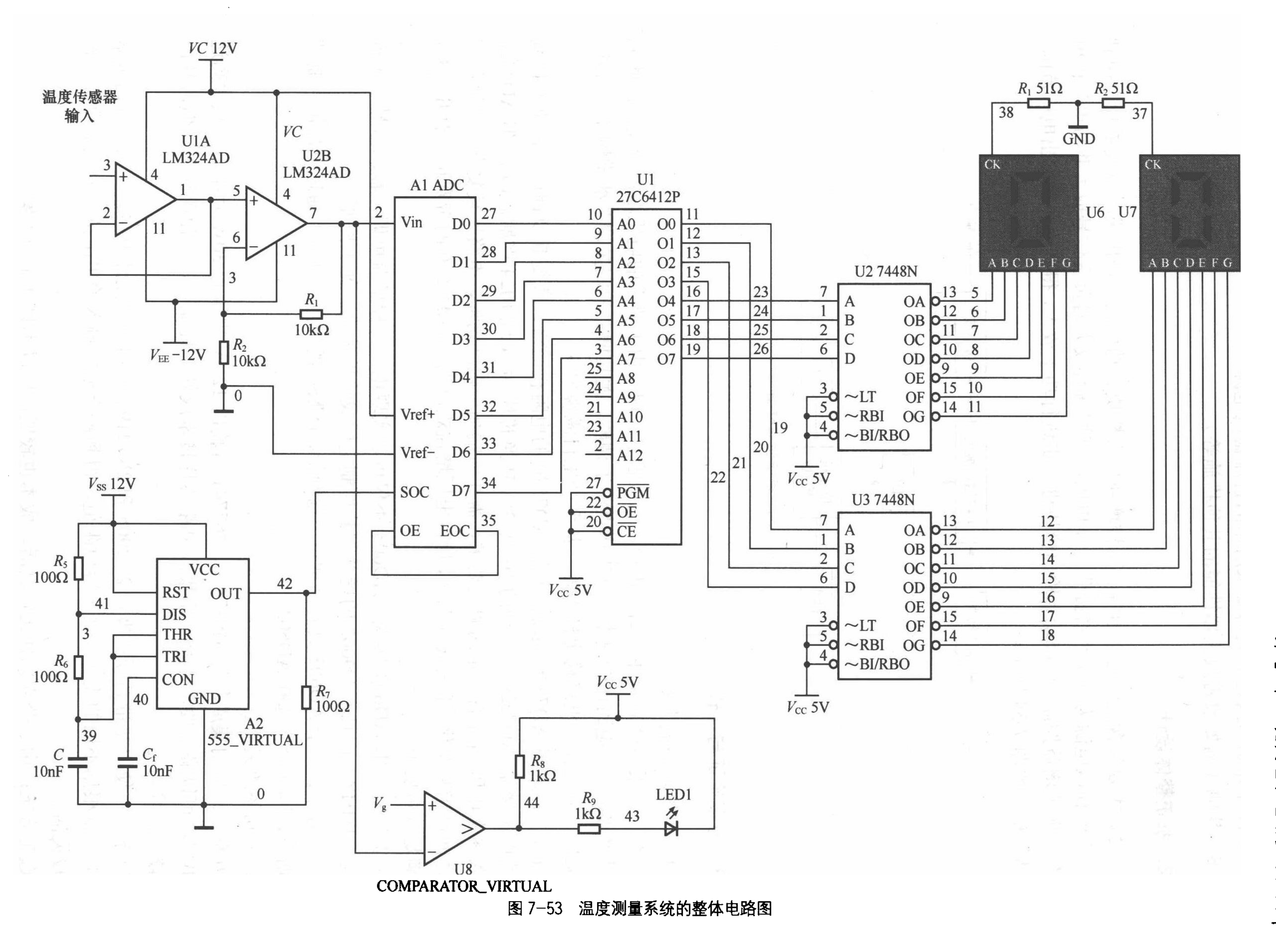


图7-53 温度测量系统的整体电路图

(7) 设计并制作满足放大器要求所用的直流稳压电源。

(8) 其他 (例如改善放大器性能的其他措施等)。

2. 系统整体设计

本设计主要包含直流稳压电源、可控增益放大模块、功率放大输出模块以及单片机控制模块。直流稳压电源为整个系统提供工作电压，可控增益放大器负责信号放大并与单片机电路配合实现了增益控制。后级功率输出模块进一步进行功率放大，得到较高的输出电压范围。整体结构框图如图 7-54 所示。

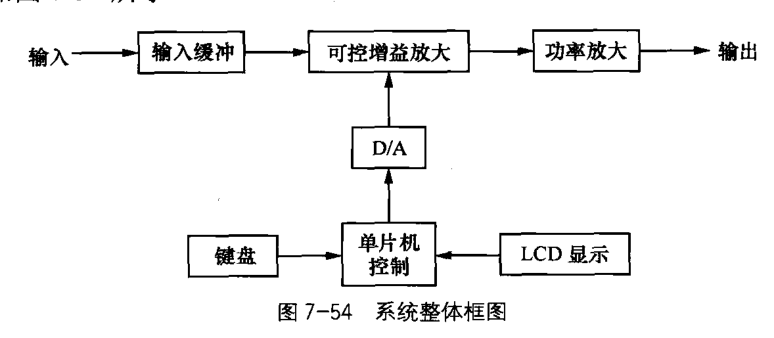


图 7-54 系统整体框图

3. 方案论证与比较

(1) 可控增益放大器。

方案一：采用分立元件（如三极管、二极管）实现。由于要求的电压增益较高，因此需要经过多级放大，电路比较复杂，且工作点难于调整，尤其增益的定量调节非常困难。另外，由于采用分立元件，电路稳定性很差，甚至产生自激现象。

方案二：采用 AD7520 实现。利用 AD7520 的电阻权网络，改变其反馈电压，进而控制电路的电压增益。AD7520 是一种的 10 位 D/A 转换芯片，可实现 1024 档增益调节。但 AD7520 对输入参考电压 V_{REF} 有一定的幅度要求，需在前端加一幅度调理，电路比较复杂；且增益和电压之间呈指数变化，不是线性关系，使得电压增益难以调节；AD7520 的带宽也只有几 kHz，不能满足频带要求。

方案三：采用 AD603 实现。AD603 不仅可以实现电压可控增益可调，输入信号的频带也可达 90MHZ；而且，AD603 的增益与其控制电压成线性关系，便于控制。

选定方案：由于 AD603 具有高增益、宽频带、低噪声的优点，且便于通过单片机控制其电压增益，因而最终选择 AD603。

(2) 功率放大器。

方案一：使用集成芯片 M67741。M67741 可输出 12.5V 电压和 30W 功率，且频率高达 135MHZ，可以满足要求。且使用集成电路芯片技术成熟、性能稳定、使用简单。但芯片的价格较贵，且功耗大。

方案二：使用宽带高频三极管 2N2905 和 2N2219。使用分立元件电路较复杂，且调试繁琐，但可以根据不同的需要改变电容电阻值，得到最合适的输入输出阻抗、放大倍数等，相对比较灵活。

选定方案：由于分立元件比较灵活，成本也较低，所以本设计采用方案二。

（3）稳压电源。

方案一：开关稳压电源。此方案效率高，但电路复杂，开关电源的工作频率通常为几十至几百 kHz，基波与很多谐波均在本放大器通频带内，极容易带来串扰。

方案二：线性稳压电源。串联型稳压电源电路简单、效率高，若采用集成三端稳压器，可以方便地得到可靠的输出电压。

选定方案：考虑开关电源成本比较高，稳定性比较好，所以本设计选择方案二。

4．理论计算与分析

（1）宽带增益积的计算。

单级 AD603 的电压增益可按公式 $G_{\text{ain}}(\text{dB})=40\times V_G+10$ 计算，其中，V_G 是差分输入电压。可见，AD603 的增益与其控制电压成线性关系。也就是说，只要用单片机控制 D/A 输出一线性变化的控制电压，便可得到线性变化的电压增益输出。假设键盘输入一预置电压增益 G_F，则 D/A 输出电压的理论值应为 $V_G=(G_F-10)/40$。

又由于 D/A 的输出电压为 $V_{\text{OUT}}=\dfrac{D_{\text{IN}}}{2^{12}}\times V_{\text{REF}}$（12 位 DA），可算出 D_{IN} 的理论值为 $D_{\text{IN}}=\dfrac{V_{\text{OUT}}}{V_{\text{REF}}}\times 2^{12}$。

因此，只要利用单片机向 D/A 送 12 位的 D_{IN}，在 D/A 的输出端便可得到所需的控制电压 V_G，从而控制 AD603 产生 G_F 大小的电压增益。

由于单级 AD603 只能实现 31.07dB 增益，而指标要求电压增益≥60dB，所以必须采用两片 AD603 级联。AD603 有两种增益连接方式：顺序级联和并联级联。为了控制精度、提高性噪比，采用顺序级联方式。

当高频信号通过两级级联的放大器时很容易产生自激，从而限制了系统的宽带增益级。为此，需要抑制自激，最终使系统的宽带增益级达到 400MdB。

（2）增益起伏控制。

由于 AGC 对不同频率的输入信号的增益不完全相同，所以会造成增益谱不平坦，即有增益起伏。设计选用的 AD603 其内部含有负反馈，可以较好地控制增益起伏。另外，在放大器的输出端对地接耦合小电容 47pF 和 2μH 电感等措施进行补偿，从而将增益起伏控制在 1dB 以下，满足指标要求。

（3）线性相位。

由于系统对不同频率的信号响应产生的延时有可能不同，所以会产生相移（落后或超前）。要使所有频率的输入具有相同的相位，则需设计线性相位滤波器，通常采用 FIR 滤波器，也可以采用模拟的方法。本设计采用电压并联负反馈，较好地抑制了相位漂移。

（4）抑制直流零点漂移。

零点漂移是指输入为零时输出端仍有缓慢变换的输出电压的现象。设计中常采用补偿的方法来抑制零点漂移。多级级联时，为防止温漂逐级递增，必须采用阻容耦合或变压器耦合。

本设计中，在放大器之前设计了一个输入缓冲级，采用低温漂的运算放大器，对输入的小信号进行适当的放大，使输入信号远大于温漂，这样温漂的影响便可忽略。同时，在功率

放大器输入端设置了调零功能。

（5）放大器的稳定性。

由于晶体管有反向传输导纳存在，会产生自激，影响放大器的稳定性。为了提高运放的稳定性，可以从电路上设法消除晶体管的反向作用，采用失配法使其单向化。

由于高频信号通过两级级联的AD603很容易产生自激，因此采取了下面几种抑制自激的方法：

① 在可控增益放大器的控制端加电感以及在其输出端加滤波电路来滤除高频噪声；

② 在功率放大器的反馈端加一小电容以防止输出的高频大信号耦合到输入端；

③ 大面积布地。

5. 电路与程序设计

（1）输入缓冲级。

由于题目要求输入电压有效值小于10mv，当输入小信号时AD603的放大性能很差，所以在输入端采用低温漂运放构成输入缓冲级，对小信号输入信号进行一定的放大，同时对后续的AGC调零。电路设计可参阅测量放大器一节。

（2）可控增益放大器。

可控增益放大器采用两级AD603级联而成，由于输入的信号包含直流分量，其电路图如图7-55所示。

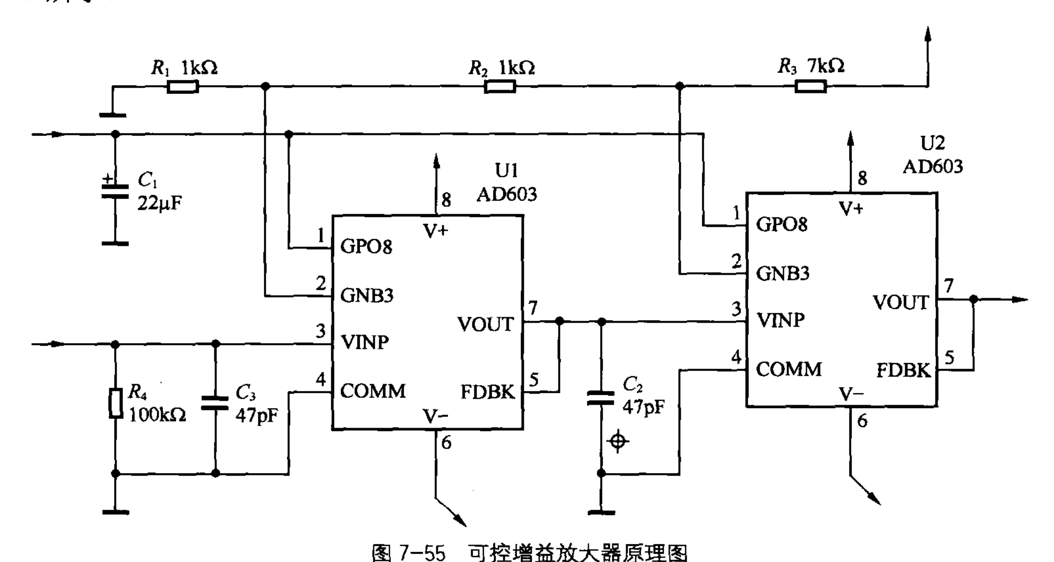


图7-55 可控增益放大器原理图

其中，+5V电压经过R_1，R_2，R_3分压，分别为两级AD603提供+0.5V和+1.5V的控制电压差，从而使电压增益值从0dB开始调节。控制电压$V_{control}$是由D/A产生的直流电压，所以在其输入端就近对地接一直流旁路电容。图7-55中的C_1即直流旁路电容，起着减小输入的高频噪声干扰的作用。C_3、C_4为去耦电容，滤除高频信号噪声和反馈引起的耦合作用。

（3）功率放大。

AD603的输出电压只有2V，而指标要求输出电压有效值高达10V，故需进行功率放

大。由于输入信号的频带比较宽，功率也较高，所以采用宽带功率放大三极管 2N2219 和 2N2905。

功放电路由两级功率放大组成，第一级将输入信号分成直流信号和高频信号，对两路分别进行电压放大，整个功放电路的电压增益主要集中在这一级上；第二级对信号进行电压合成和电流放大，提高负载驱动能力。另外，功放电路还采用负反馈以提高带宽，以满足指标的频带要求。功放电路原理图如图 7-56 所示。

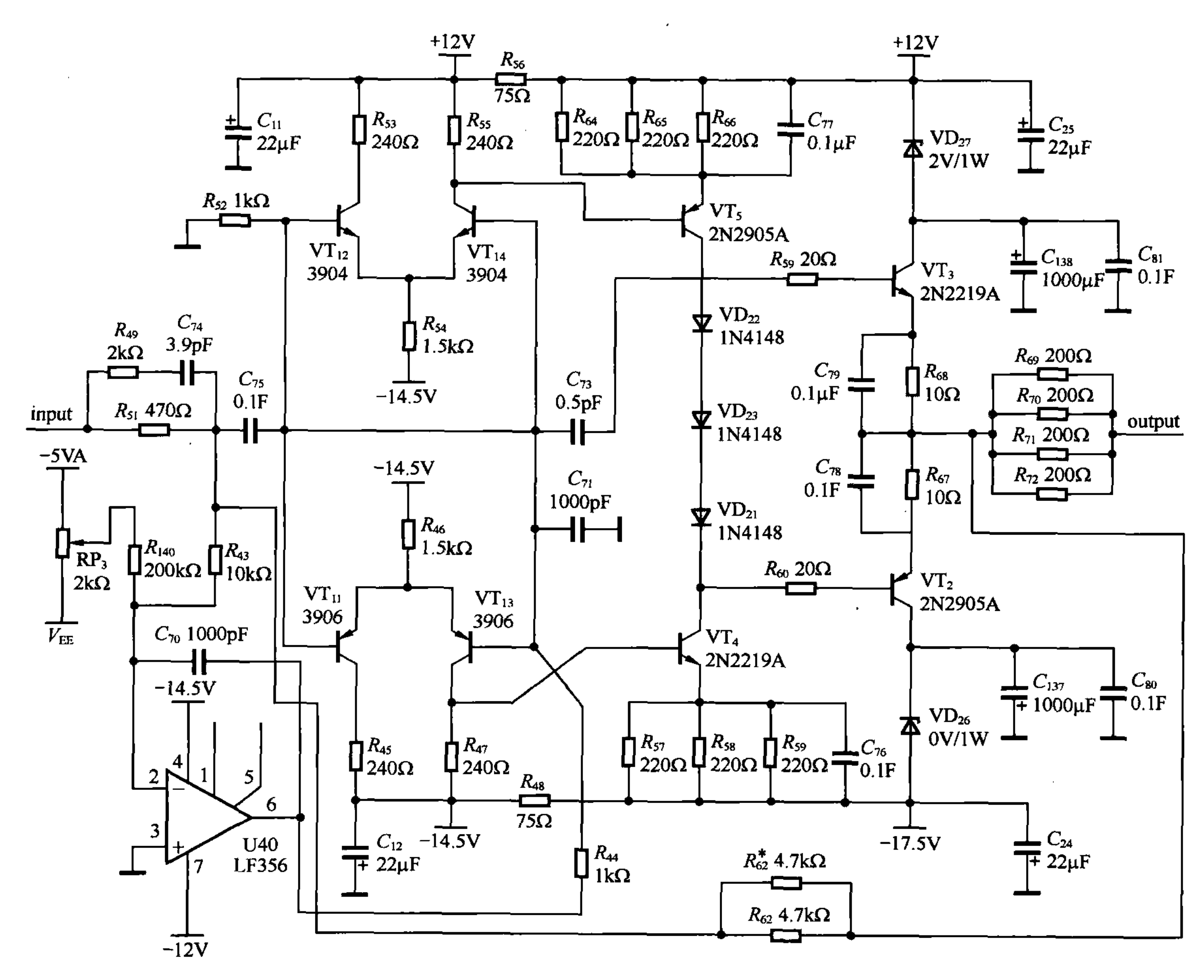


图 7-56　功率放大器原理图

（4）增益控制。

控制部分主要由单片机和 D/A 组成。设计采用 ST89S51 和 12 位的 D/A TLV5616，单片机首先接收键盘的预置增益值，一方面送给 LCD 显示，另一方面通过一定的运算后输入 12 位 D/A，产生控制电压控制 AD603 自动调节增益值，输出所需的信号。其框图如图 7-57 所示。

（5）直流稳压电源。

直流稳压电源主要包括 EMI 滤波、变压器、桥式全波整流和三端稳压管几个模块。设计中包括两个重点。

键盘 → ST89S51 → TLV5616 → AD603 → 输出
ST89S51 → LCD

图 7-57　增益控制框图

① 由于电源整流电路中的充放电电容存在充电及放电时间之分，所以必然会有纹波存在。数模转换部分受电源纹波的影响较大，经过稳压器的电源一定要滤掉电源纹波后才可以加到 DAC 的电源输入端。通常的做法是在电源和相应的地之间加几个不等值的滤波电容，以滤掉不同频率的纹波。在本次设计中采用了 EMI 滤波。

② 选用的稳压芯片不同，经过其产生的压降也不同，得到的输出电压也不同。由于前级和后级放大器需要+10V 和+30V 电压，而单片机工作在+5V 电压，所以采用输出可变的稳压芯片（LM317T），分别输出±5V，±12V 和±18V 的电压。

稳压电源的框图如图 7-58 所示。

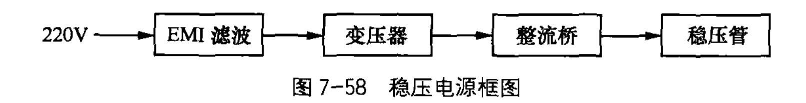


图 7-58　稳压电源框图

220V 的交流输入经过 EMI 滤波后，送到变压器的初级线圈；次级输出的电压再经过整流桥，得到稳定的直流电压；然后经过三端稳压管 LM317T，输出所需的电压。在 LM317T 的 ADJ 端加一个接地的滤波电容，会使纹波抑制比大幅度地提高，从而为高频小信号运算放大器提供相当稳定的电压。

（6）程序设计。

软件设计比较简单，主要包括单片机、D/A 和液晶显示。采用单片机 C 语言，分别对单片机和液晶初始化，并根据预置增益值进行一定的运算，驱动 D/A 产生所需的控制电压。其程序流程图如图 7-59 所示。

6. 测试方案及测试结果

（1）测试方案及条件。

① 测试仪器：信号发生器：DG1022；示波器：TDS210；扫频仪：BT3C-A。

② 测试方案。

系统带宽和增益的测量主要有静态测试法和动态测试法。两者均利用信号发生器产生输入信号，前者采用示波器/交、直流电压表测试其静态特性，后者采用扫频仪测量其幅频特性。系统的测试框图如图 7-60 所示。

（2）测试结果。

① 带宽和增益的测量。

利用信号发生器产生输入信号送入到宽带直流放大器中，然后从键盘输入预置的增益值，用示波器/交、直流电压表分别测量其输入和输出端的电压有效值；不断减小输入信号的电压有效值，并增大其频率，重复测量。首先设定增益为 20dB，测得其宽带数据表如表 7-5 所示。

表 7-5 增益为 20dB 时放大器的测试数据表

U_i/mV \ U_o/mV \ f	0Hz	100Hz	10kHz	100kHz	1MHz	5MHz	8MHz	10MHz
16	15.90	15.88	15.85	15.76	15.72	15.70	15.68	10.28
28	27.82	27.80	27.60	27.50	27.500	27.300	27.30	18.45
40	39.90	39.80	39.80	39.60	39.50	39.50	39.40	23.40
58	47.83	47.80	47.65	47.60	47.59	47.56	47.50	31.40
70	69.70	69.76	69.71	69.65	69.62	69.58	69.56	51.56
100	99.81	99.76	99.72	99.67	99.64	99.61	99.58	79.57

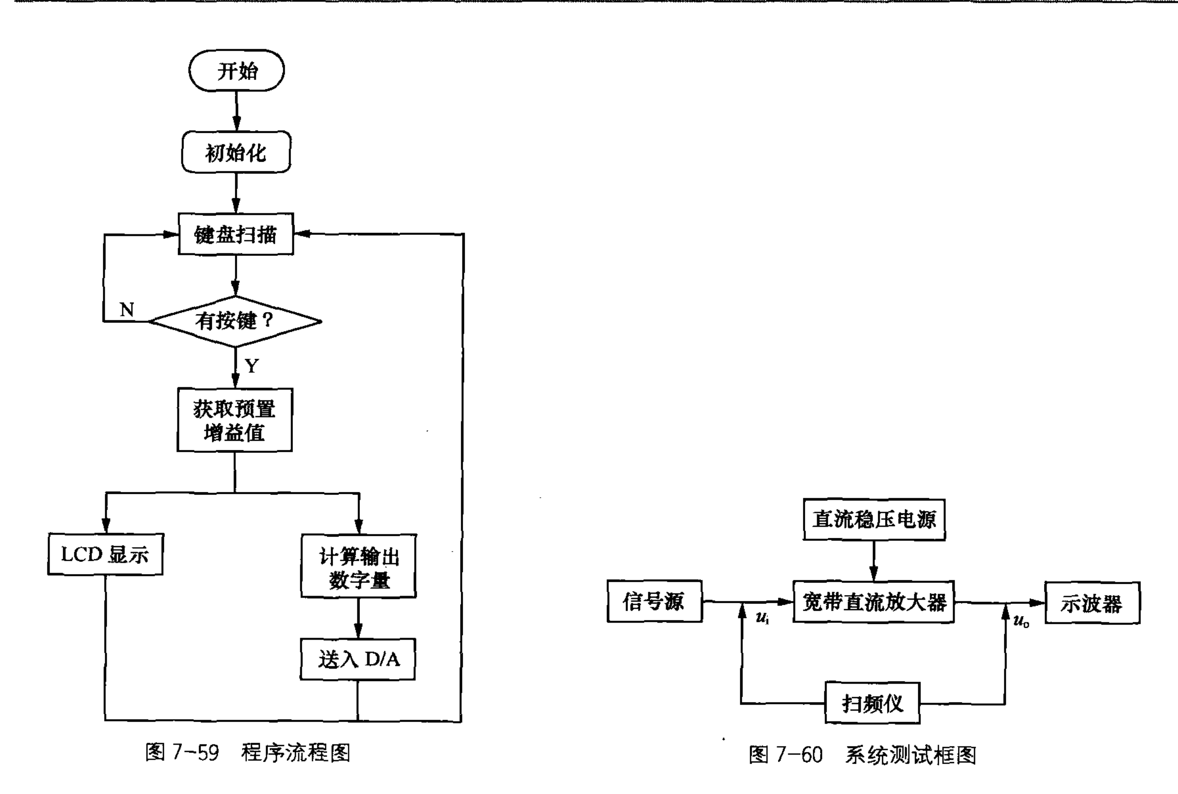


图 7-59 程序流程图

图 7-60 系统测试框图

另外，改变输入信号的有效值，测量其可调增益值如表 7-6 所示。

表 7-6 增益测试数据表

U_i/mV \ U_o/mV \ dB	0	5	10	20	30	40	50	55
16	15.90	28.27	50.24	156.58	498.72	1572.7	5022.5	8940.1
28	27.82	49.21	87.84	275.20	881.95	2771.8	8897.4	8783.4
40	39.90	70.62	126.03	390.60	1264.2	3884.5	8785.2	8844.2
58	47.83	85.13	150.60	470.32	1523.6	4722.3	8981.3	8792.7

② 放大器输入阻抗的测量。

输入阻抗的测量框图如图 7-61 所示。假设放大器输入阻抗为 R_1，在其两端加一信号源 U_s，然后在输入端和信号源之间串接一电阻 R_s，再测 R_1 两端的电压，设为 U_i，则 $R_i=\frac{R_s\times U_i}{U_s-U_i}$。测试可得设计的放大器输入阻抗为 100Ω。

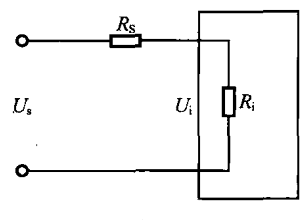


图 7-61　输入阻抗测量图

③ 负载阻抗的测量。经万用表测得负载阻抗为 51Ω。

（3）测试结果分析。

由表 7-5 可知，放大器的放大倍数同理论值存在的误差基本相同。当输入信号一定时，放大器对不同频率的输入信号放大倍数相同，增益平坦度满足＜1dB 的要求。系统带宽可达 8MHz，输入信号最低可至 4mV，输出电压的有效值也可高达 10V，指标全部实现。

7．总结

本设计采用两级 AD603 构成主要的增益控制模块，再辅以功率放大器和单片机控制，可对低至几毫伏的输入电压进行放大，电压增益可达 50dB，通频带也可做到 8MHz，输出电压有效值最大达 10V，指标全部达到。

习　题　七

1．简述现代电子电路设计的基本步骤。

2．观察、分析常用电器所存在的不足之处，提出相应的改进方案。

3．设计一种无线心音传感器，可以通过计算机或手机接收信号，并在屏幕上显示心音图。

参 考 文 献

[1] 臧春华，郑步生，魏小龙. 电子设计自动化技术[M]. 北京：机械工业出版社，2004.
[2] 王莲英. 基于 Multisim10 的电子仿真实验与设计[M]. 北京：北京邮电大学出版社，2009.
[3] 刘刚，王立香，任鲁涌. Multisim10&Ultiboard10 原理图与 PCB 设计[M]. 北京：电子工业出版社，2009.
[4] 张新喜，许军，王新忠，杨雨迎. Multisim10 电路仿真及应用[M]. 北京：机械工业出版社，2010.
[5] 张豫滇，谢劲草. 电工电子实验技术（下册）[M]. 南京：河海大学出版社，2006.
[6] 黄培根，任清褒. Multisim10 计算机虚拟仿真实验室[M]. 北京：电子工业出版社，2008.
[7] 唐亚平. 电子设计自动化（EDA）技术[M]. 北京：化学工业出版社，2002.
[8] 叶建波，余志强. Protel 99 SE & EWB 5.0[M]. 北京：清华大学出版社，2005.
[9] 张敬怀. EDA 技术与电子系统工程设计[M]. 北京：中国铁道出版社，2003.
[10] 欧阳知建，吉兵，李旭平. 电子装配实践教程[M]. 南京：河海大学出版社，2004.
[11] 赵月飞，郭会平，胡仁嘉. Protel 99 SE 基础与实例教程[M]. 北京：机械工业出版社，2009.
[12] 丁剑冰. 基于手机平台的心音信号提取与处理. 南京：南京邮电大学，2009.
[13] 郭锁利，刘延飞，李琪，王晓戎，常春藤. 基于 Multisim9 的电子系统设计、仿真与综合应用[M]. 北京：人民邮电出版社，2008.
[14] 王志功，景为平. 集成电路设计与 EDA 工具应用[M]. 1 版. 南京：东南大学出版社，2004.
[15] 姜艳波. 数字集成电路应用百例[M]. 1 版. 北京：化学工业出版社，2009.
[16] 何小虎，胡庆生，肖洁. 深亚微米下 ASIC 后端设计及实例[J]. 中国集成电路. 2006，(8)：45-48.
[17] 刘刚. LabVIEW 8.20 中文版编程及应用[M]. 北京：电子工业出版社，2008.
[18] 雷永. 虚拟仪器设计与实践[M]. 北京：电子工业出版社，2005.
[19] Gary W Godson. LabVIEW Graphic Programming[M]. USA：MC Graw-Hill，1998.
[20] 魏海燕，杨建新. 基于 LabVIEW 的虚拟仪器开发[J]. 机械工程师，2000，(4)：50-51.
[21] 成谢锋，陶冶薇，张少白. 基于 WPT 分层和独立子波函数的单路心音混合信号 BSS 新方法[J]. 南京邮电大学学报（自然科学版），2009，4（29）：50-55.
[22] 成谢锋，李能禾. 心音虚拟仪器中信号的检测与处理[J]，仪器仪表学报，2009，10 I（30）：96-99.
[23] 杨帮文. 新型集成器件实用电路（修订版）[M].北京：电子工业出版社，2006.
[24] 周子昂，王福源，魏军辉. 基于 FPGA 的通用分频器设计[M]. 郑州：郑州大学，2008.
[25] 欧伟民. 可编程的十字路口交通信号灯控制电路[J]. 现代电子技术，2001，(10)：25～28.
[26] 林根远. 多用手写字符输入板[M]. 北京耀辉电子通信公司，2006.
[27] 贾智平，张瑞华. 嵌入式系统原理与接口技术[M]. 北京：清华大学出版社，2005.

[28] 谢宜仁. 单片机硬件接口电路及完例解析[M]. 北京：电子工业出版社，2009.
[29] 马明建. 数据采集与处理技术[M]. 西安：西安交通大学出版社，2005.
[30] 李联富，刘飞. 基于 S3C2410 的无线数据采集系统[J]. 现代电子技术，2009，24（311），171～172.
[31] 陈赜，邹道胜. 电子创新设计技术[M]. 北京：科学出版社，2008.
[32] 孙肖子，邓建国，陈南. 电子设计指南[M]. 北京：高等教育出版社，2006.
[33] 张新喜，许军，王新忠，杨雨迎. Multisim 10 电路仿真及应用[M]. 北京：机械工业出版社，2010.
[34] 聂典，丁伟. 基于 Multisim 10 的 51 单片机仿真实战教程——使用汇编和 C 语言[M]. 北京：电子工业出版社，2010.